EECE 5639 Computer Vision I

Lecture 22

Object Recognition: Eigenimages, Bag of Words

Project 4 is out. Now Due April 16.

Hw 5 is out. Now Due April 19

Next Class

Object Recognition: deep learning

PCA Theorem

Let x_1 x_2 … x_n be a set of n N^2 x 1 vectors and let $\bar{x}$ be their average:

$$\mathbf{x}_i = \begin{bmatrix} x_{i1} \\ x_{i2} \\ \vdots \\ x_{iN^2} \end{bmatrix} \qquad \bar{\mathbf{x}} = \frac{1}{n} \sum_{i=1}^{i=n} \begin{bmatrix} x_{i1} \\ x_{i2} \\ \vdots \\ x_{iN^2} \end{bmatrix}$$

Note: Each N x N image template can be represented as a N^2 x 1 vector whose elements are the template pixel values.

PCA Theorem

Let X be the N^2 x n matrix:

$$X = \left[\begin{array}{cccc} \mathbf{x}_1 - \bar{\mathbf{x}} & \mathbf{x}_2 - \bar{\mathbf{x}} & \cdots & \mathbf{x}_n - \bar{\mathbf{x}} \end{array} \right]$$

Note: subtracting the mean is equivalent to translating the coordinate system to the location of the mean.

PCA Theorem

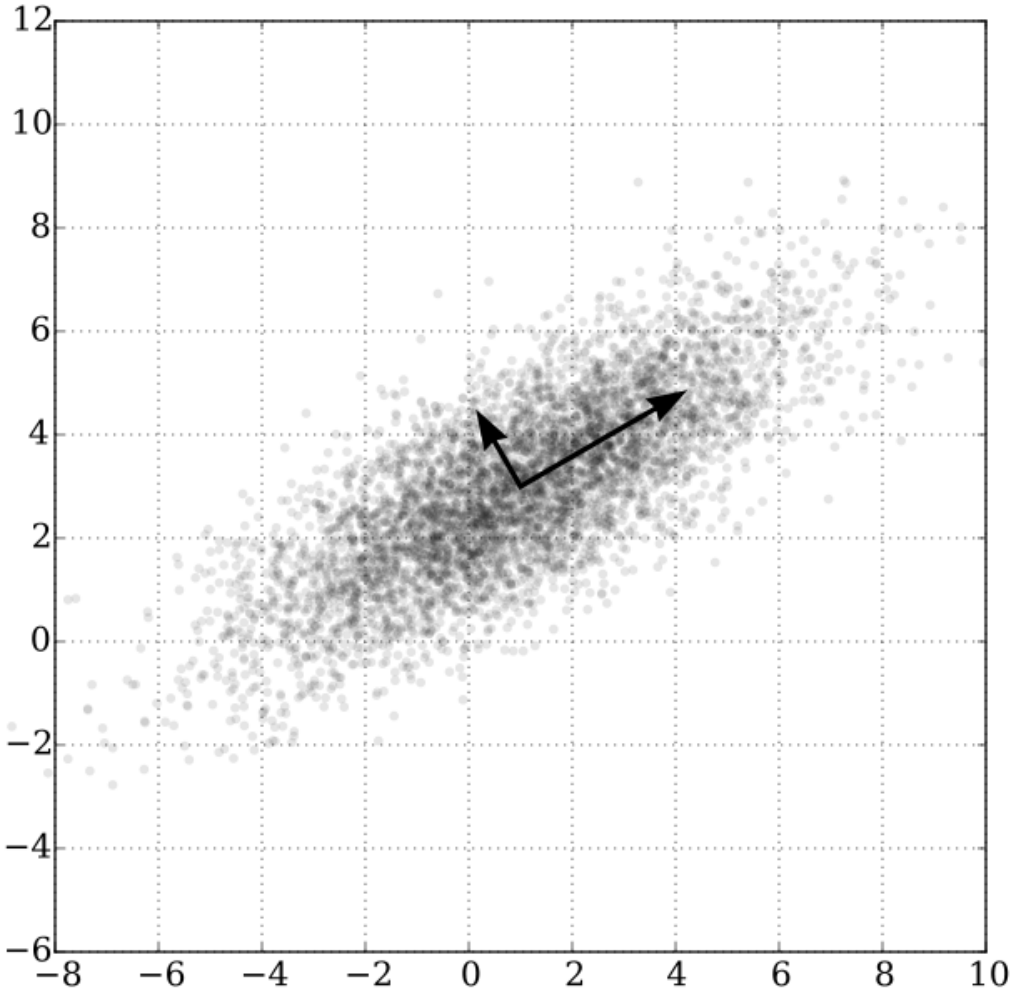
Let Q = X X^T be the N^2 x N^2 matrix:

$$Q = XX^T = \left[\begin{array}{cccc} \mathbf{x}_1 - \bar{\mathbf{x}} & \mathbf{x}_2 - \bar{\mathbf{x}} & \cdots & \mathbf{x}_n - \bar{\mathbf{x}} \end{array} \right] \left[\begin{array}{c} (\mathbf{x}_1 - \bar{\mathbf{x}})^T \\ (\mathbf{x}_2 - \bar{\mathbf{x}})^T \\ \vdots \\ (\mathbf{x}_n - \bar{\mathbf{x}})^T \end{array} \right]$$

Notes:

1. Q is square
2. Q is symmetric
3. Q is the ***covariance*** matrix
4. Q can be very large (remember that N^2 is the number of pixels in the template)

PCA



PCA Theorem

Theorem:

Each x_j can be written as: $\mathbf{x}_j = \bar{\mathbf{x}} + \sum_{i=1}^{i=n} g_{ji}\mathbf{e}_i$

where e_i are the n eigenvectors of Q with non-zero eigenvalues.

Notes:

1. The eigenvectors e_1 e_2 … e_n span an ***eigenspace***
2. e_1 e_2 … e_n are N^2 x 1 orthonormal vectors (N x N images).
3. The scalars g_{ji} are the coordinates of x_j in the space.
4. $$g_{ji} = (\mathbf{x}_j - \bar{\mathbf{x}}).\mathbf{e}_i$$

Using PCA to Compress Data

Sort the eigenvectors e_i according to their eigenvalue:

$$\lambda_1 \geq \lambda_2 \geq \ldots \lambda_n$$

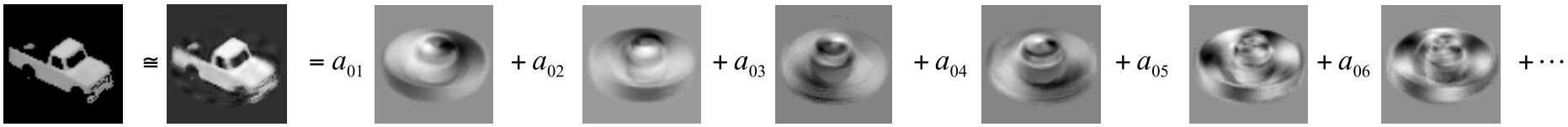
- Assuming that $\lambda_i \approx 0 \text{ if } i > k$

- Then

$$\mathbf{x}_j \approx \bar{\mathbf{x}} + \sum_{i=1}^{i=k} g_{ji}\mathbf{e}_i$$

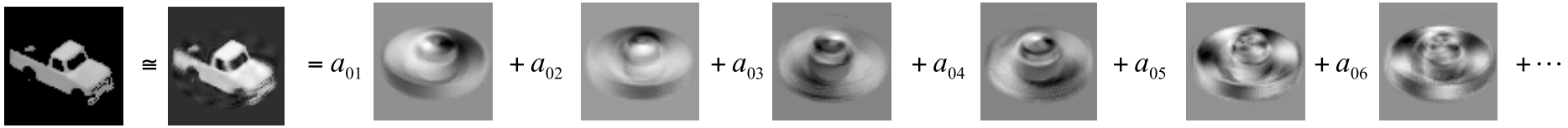
Eigenspaces: Efficient Image Storage

- Use PCA to compress the data:
  - each image is stored as a k-dimensional vector
  - Need to store k N x N eigenvectors
- $k << n << N^2$



Eigenspaces: Efficient Image Comparison

- Use the same procedure to compress the given image to a k-dimensional vector.
- Compare the compressed vectors:
  - Dot product of k-dimensional vectors
  - $k << n << N^2$



Implementing PCA

Need to find "first" k eigenvectors of Q:

$$Q = XX^T = \left[\begin{array}{cccc} \mathbf{x}_1 - \bar{\mathbf{x}} & \mathbf{x}_2 - \bar{\mathbf{x}} & \cdots & \mathbf{x}_n - \bar{\mathbf{x}} \end{array} \right] \left[\begin{array}{c} (\mathbf{x}_1 - \bar{\mathbf{x}})^T \\ (\mathbf{x}_2 - \bar{\mathbf{x}})^T \\ \vdots \\ (\mathbf{x}_n - \bar{\mathbf{x}})^T \end{array} \right]$$

Q is N^2 x N^2 where N^2 is the number of pixels in each image.
For a 256 x 256 image, N^2 = 65536 !!

Finding ev of Q

$Q=XX^T$ is very large. Instead, consider the matrix $P=X^TX$

- Q and P are both symmetric, but $Q \neq P^T$
- Q is N^2 x N^2, P is n x n
- n is the number of training images, typically $n << N$

Finding ev of Q

Let e be an eigenvector of P with eigenvalue λ:

$$Pe = \lambda e$$
$$X^T X e = \lambda e$$
$$X X^T X e = \lambda X e$$
$$Q(Xe) = \lambda(Xe)$$

Xe is an eigenvector of Q also with eigenvalue λ!

Singular Value Decomposition (SVD)

Any m x n matrix X can be written as the product of 3 matrices:

$$X = UDV^T$$

Where:

- U is m x m and its columns are orthonormal vectors
- V is n x n and its columns are orthonormal vectors
- D is m x n diagonal and its diagonal elements are called the singular values of X, and are such that:

$$\sigma_1 \geq \sigma_2 \geq \ldots \sigma_n \geq 0$$

SVD Properties

$$X = U D V^T$$

- The columns of U are the eigenvectors of $Q = XX^T$
- The columns of V are the eigenvectors of $P = X^TX$
- The squares of the diagonal elements of D are the eigenvalues of XX^T and X^TX

Algorithm EIGENSPACE_LEARN

Assumptions:

1. Each image contains one object only.
2. Objects are imaged by a fixed camera .
3. Images are normalized in size N x N:

   The image frame is the minimum rectangle enclosing the object.
4. Energy of pixels values is normalized to 1:

   $\sum_i \sum_j I(i,j)^2 = 1$
5. The object is completely visible and unoccluded in all images.

Algorithm EIGENSPACE_LEARN

Getting the data:

For each object o to be represented, o = 1, …,O

1. Place o on a turntable, acquire a set of n images by rotating the table in increments of $360^o/n$
2. For each image p, p = 1, …, n:
   1. Segment o from the background
   2. Normalize the image size and energy
   3. Arrange the pixels as vectors $\mathbf{x}^o_p$

Algorithm EIGENSPACE_LEARN

Storing the data:

1. Find the average image vector $\bar{\mathbf{x}} = \frac{1}{n.o} \sum_{o=1}^{O} \sum_{p=1}^{n} x_p^o$

2. Assemble the matrix X:

$$X = \left[\begin{array}{cccc} \mathbf{x}_1^1 - \bar{\mathbf{x}} & \mathbf{x}_2^1 - \bar{\mathbf{x}} & \cdots & \mathbf{x}_n^o - \bar{\mathbf{x}} \end{array} \right]$$

3. Find the first k eigenvectors of XX^T: $e_1,\ldots,e_k$
(use X^TX or SVD)

4. For each object o, each image p:

- Compute the corresponding k-dimensional point:

$$\mathbf{g}_p^o = \left[\begin{array}{cccc} \mathbf{e}_1 & \mathbf{e}_2 & \cdots & \mathbf{e}_k \end{array} \right] (\mathbf{x}_p^o - \bar{\mathbf{x}})$$

Algorithm EIGENSPACE_IDENTIF

Recognizing an object from the DB:

1. Given an image, segment the object from the background

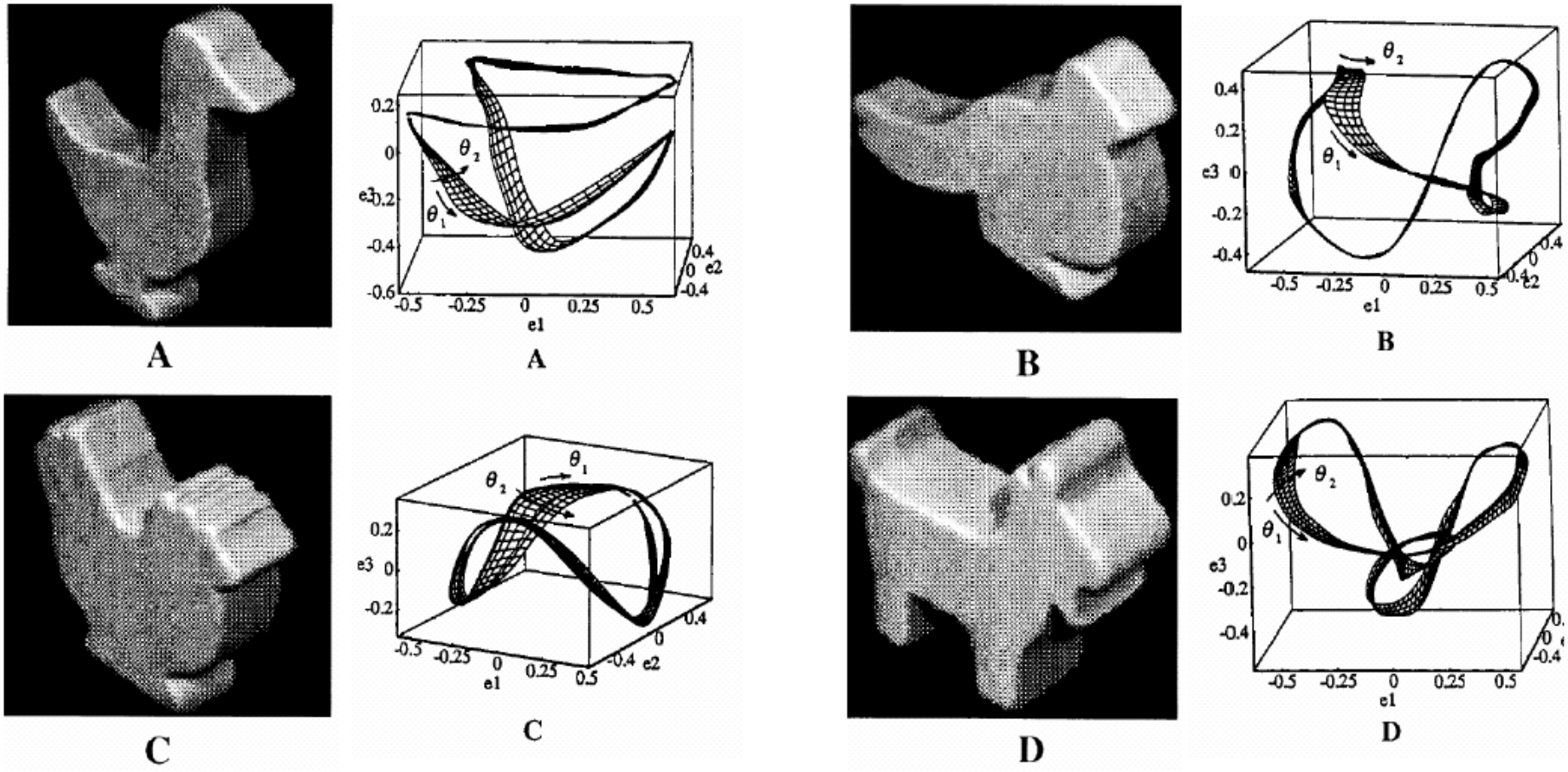
2. Normalize the size an energy, write it as a vector **i**

3. Compute the corresponding k-dimensional point:

$$\mathbf{g} = \left[\begin{array}{cccc} \mathbf{e}_1 & \mathbf{e}_2 & \cdots & \mathbf{e}_k \end{array} \right] (\mathbf{i} - \bar{\mathbf{x}})$$

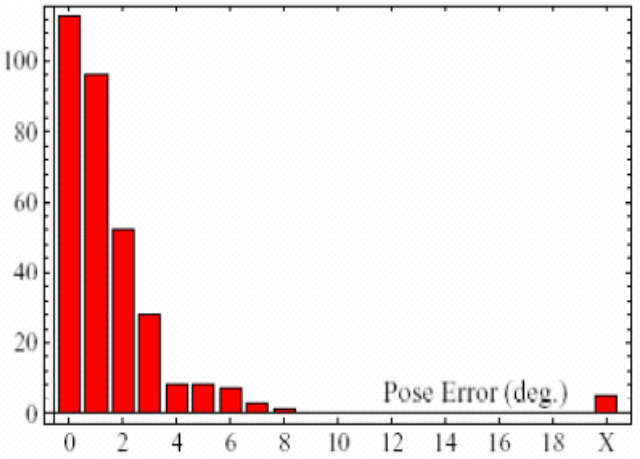
4. Find the closest $\boldsymbol{g}^{o}_{p}$ k-dimensional point to ***g***

Appearance Manifolds



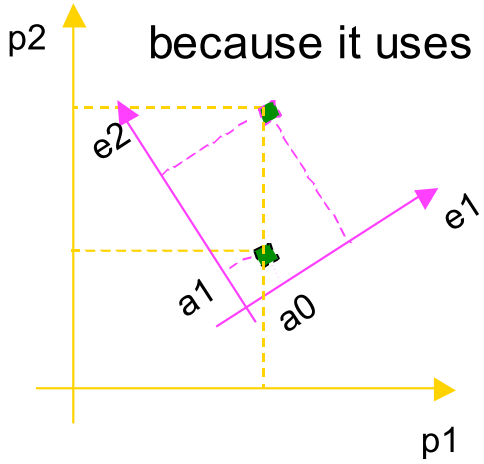
[Murase & Nayar, 1996]

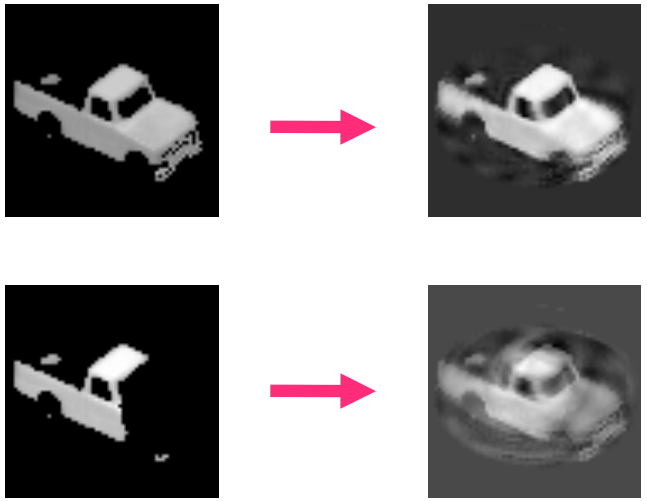
Appearance Manifolds



Murase & Nayar

PCA has problems with occlusion





Example 3: ABPs and ABRs

Use local appearance:

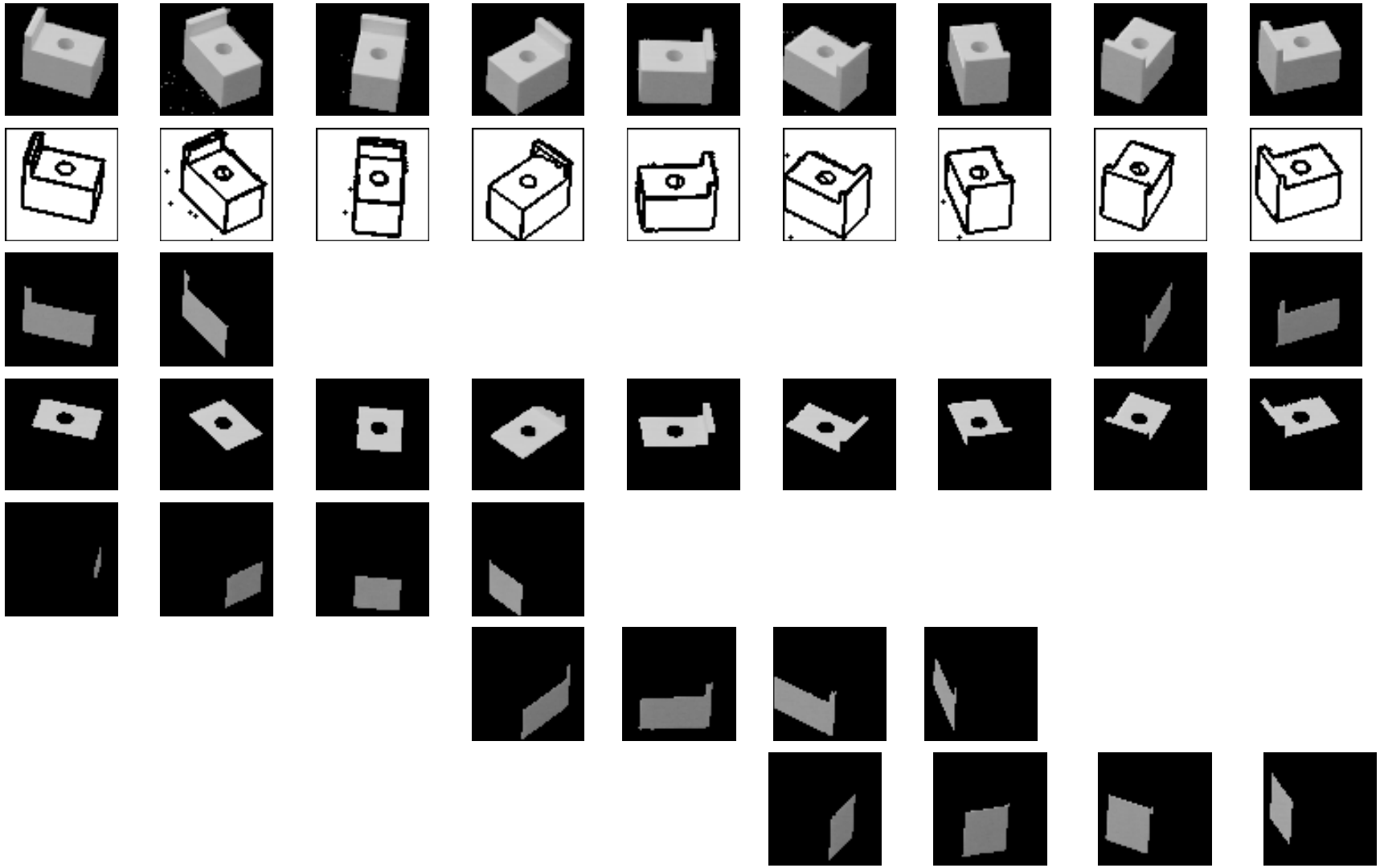
PARTS and RELATIONS

- Can model general-form objects
- Can deal with occlusion and clutter
- Can handle translation and scaling
- It is robust to segmentation problems

Parts from Images

PARTS are image regions segmented using some segmentation algorithm.

Appearance-Based Parts



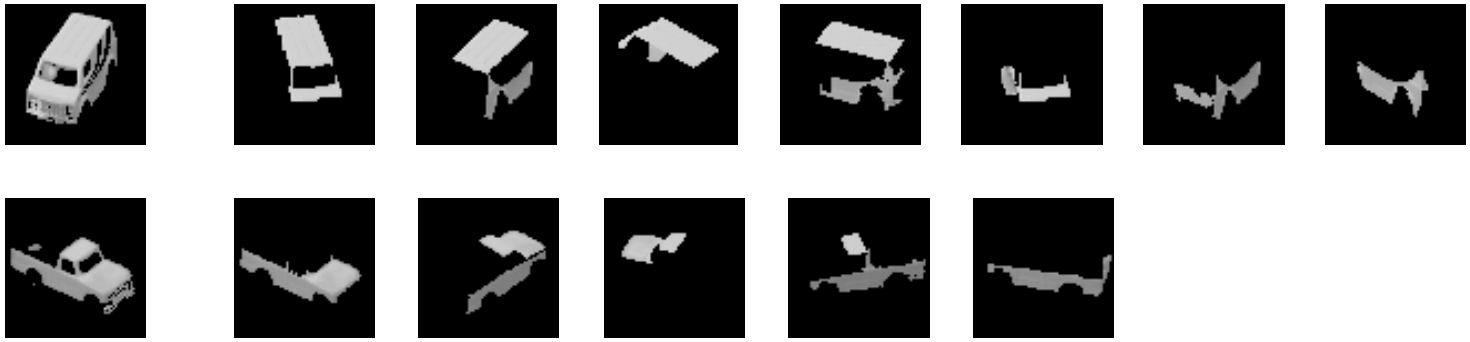
Appearance-Based Relations

Several objects can have similar parts

Use spatial relations to discriminate

Represent relations using PCA

Examples of ABRs



Recognition in cluttered scenes

Segment the given image

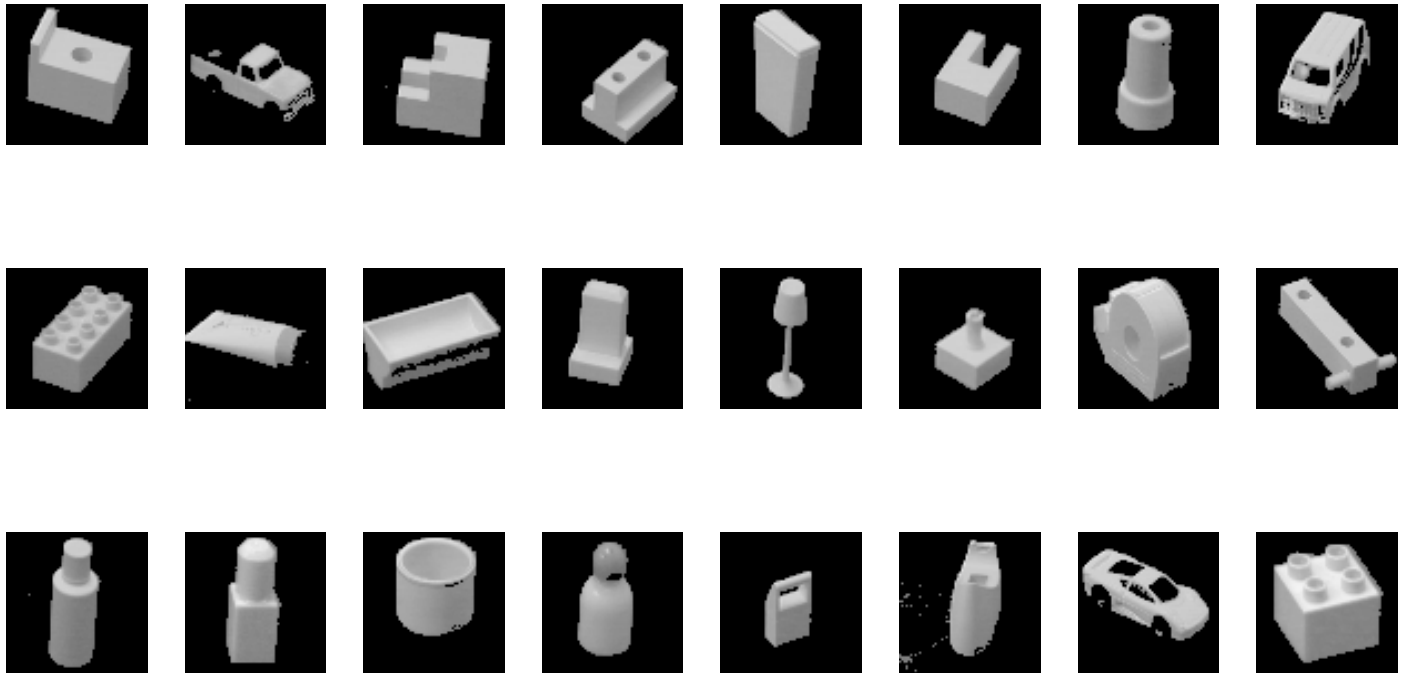
PARTS:

- Project regions into the ABP eigenspace
- Make hypotheses if $d < T1$

RELATIONS:

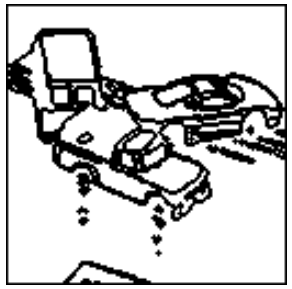
- Project adjacent regions with $T1 < d < T2$ into the ABR eigenspace
- Make hypotheses if $d' < T3$

Object Database

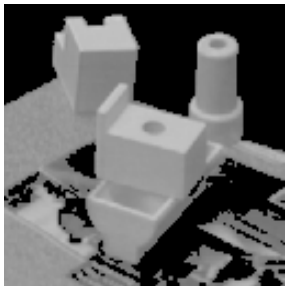
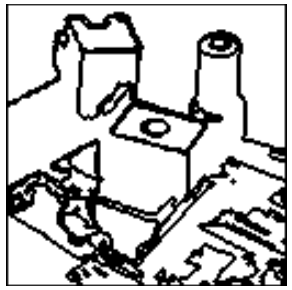


24 objects; 110 ABPs (69 shape groups/3 levels); 130 ABRs

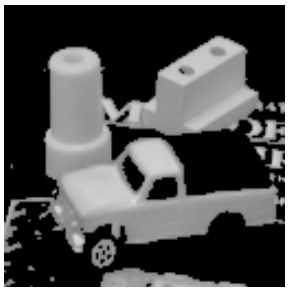
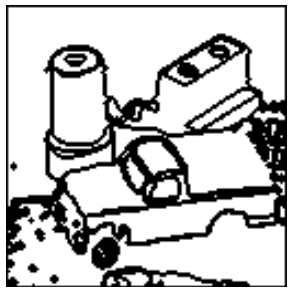
Some Examples

Identity	Score	Pose
Sport Car	0.61	302.77
Truck	0.19	4.29
Ambulance	0.10	11.52

Identity	Score	Pose
Ccube	1.36	41.12
Stamp	0.89	0.64
Holecube	0.89	14.47
Sink	0.68	152.94

Identity	Score	Pose
Stamp	1.91	93.66
Twohole	0.96	181.63
Truck	0.4	93.67

Bag of Words Models

Sivic, Russell, Freeman, Zisserman, ICCV 2005

Fei-Fei and Perona, CVPR 2005

Bosch, Zisserman, Munoz, ECCV 2006

Visual Words

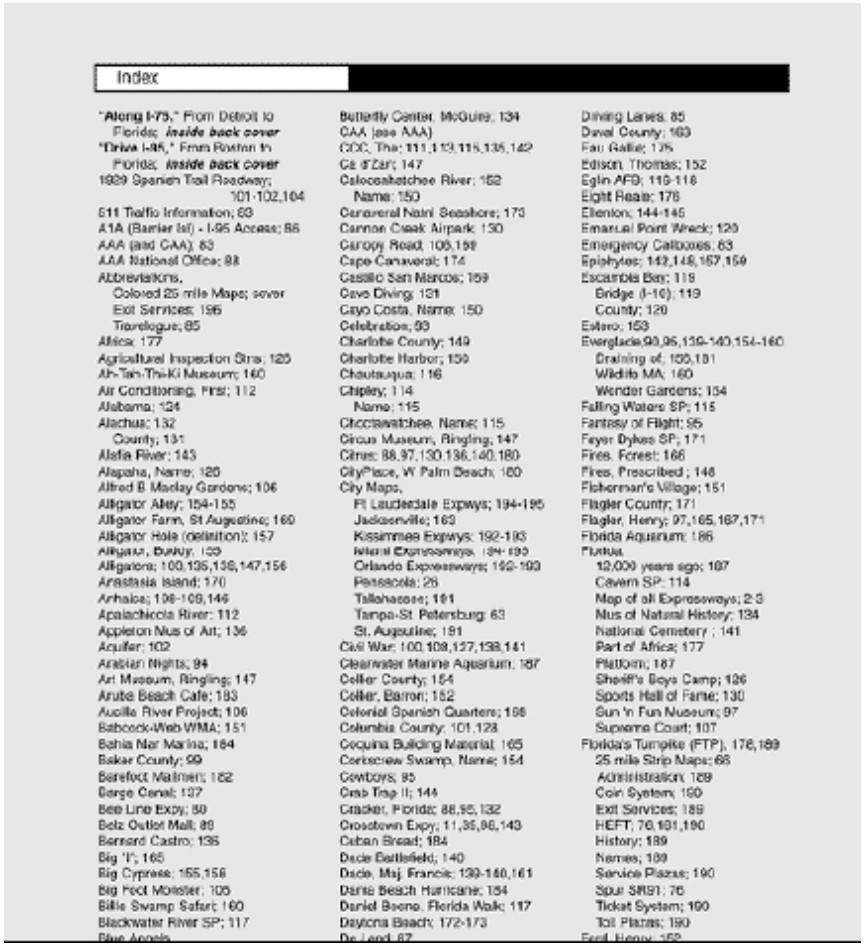
Main idea:

think of image search as a “text retrieval” problem.

Indexing Local Feature: Inverted File Index

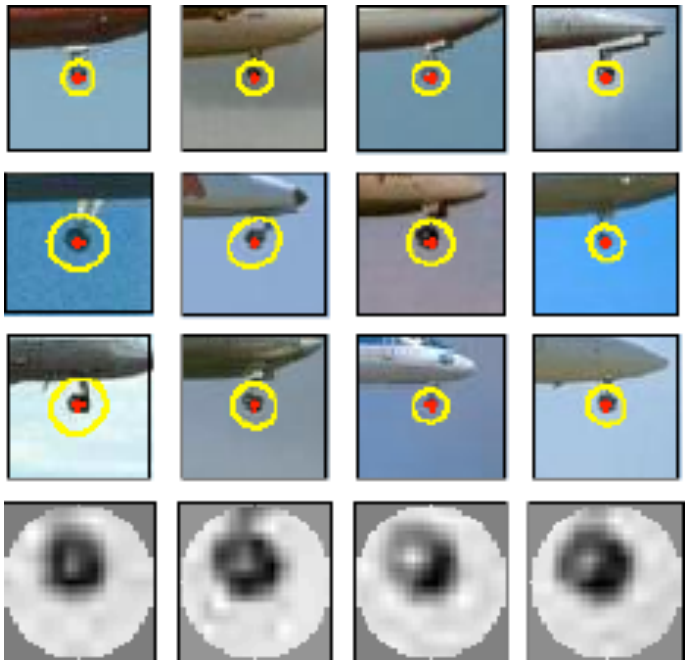
In text documents, we can quickly find all the pages where a word occurs by using an index.

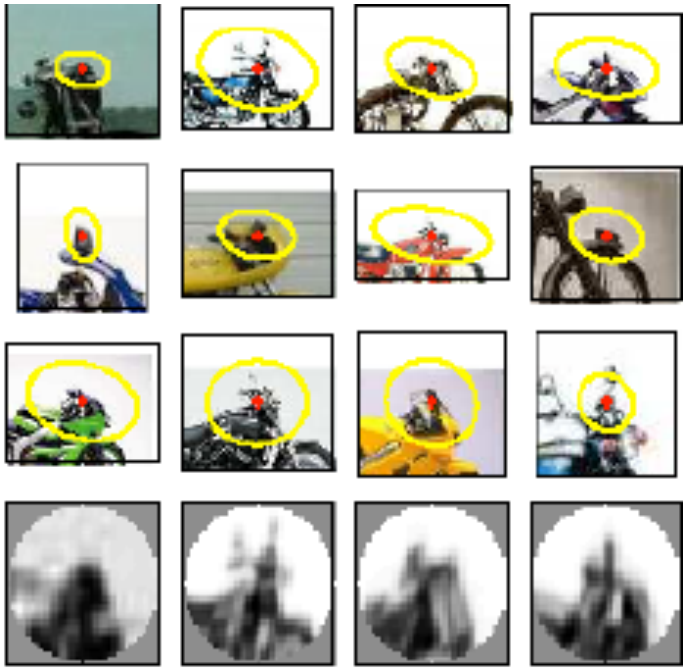
For images, we want to find all images where a feature occurs.



Index

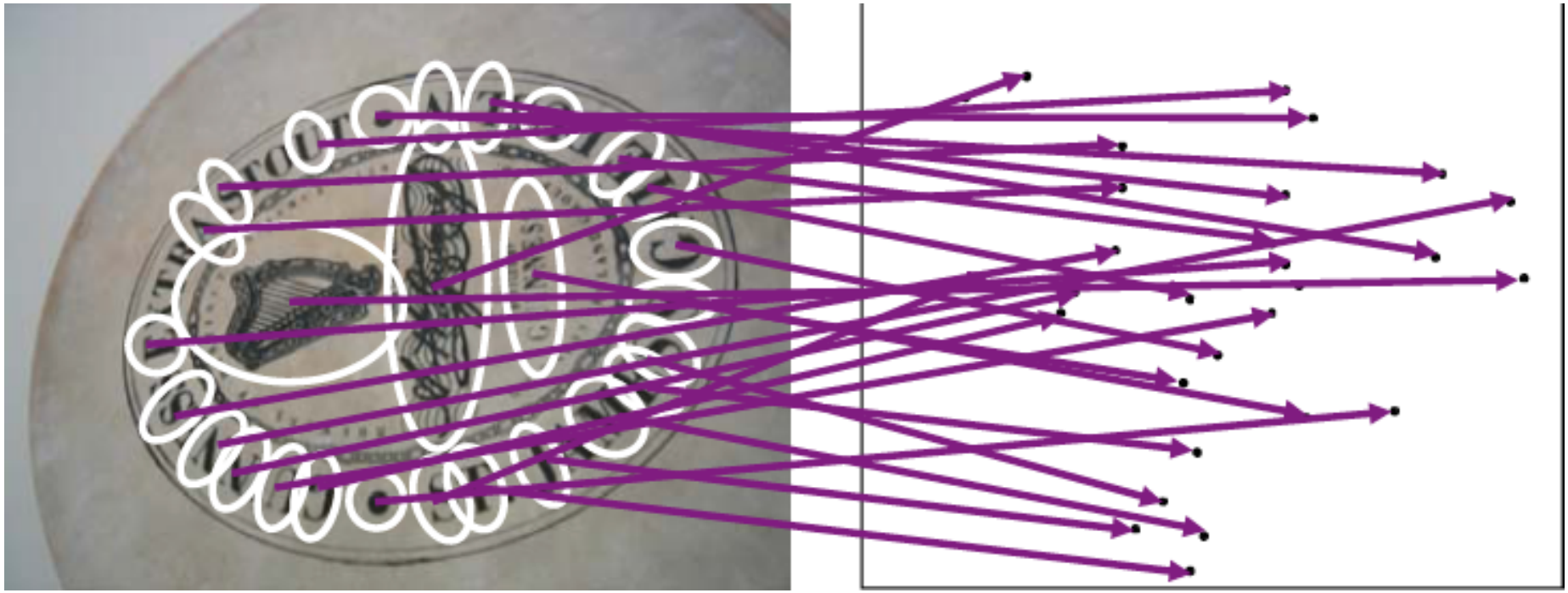
Visual Words





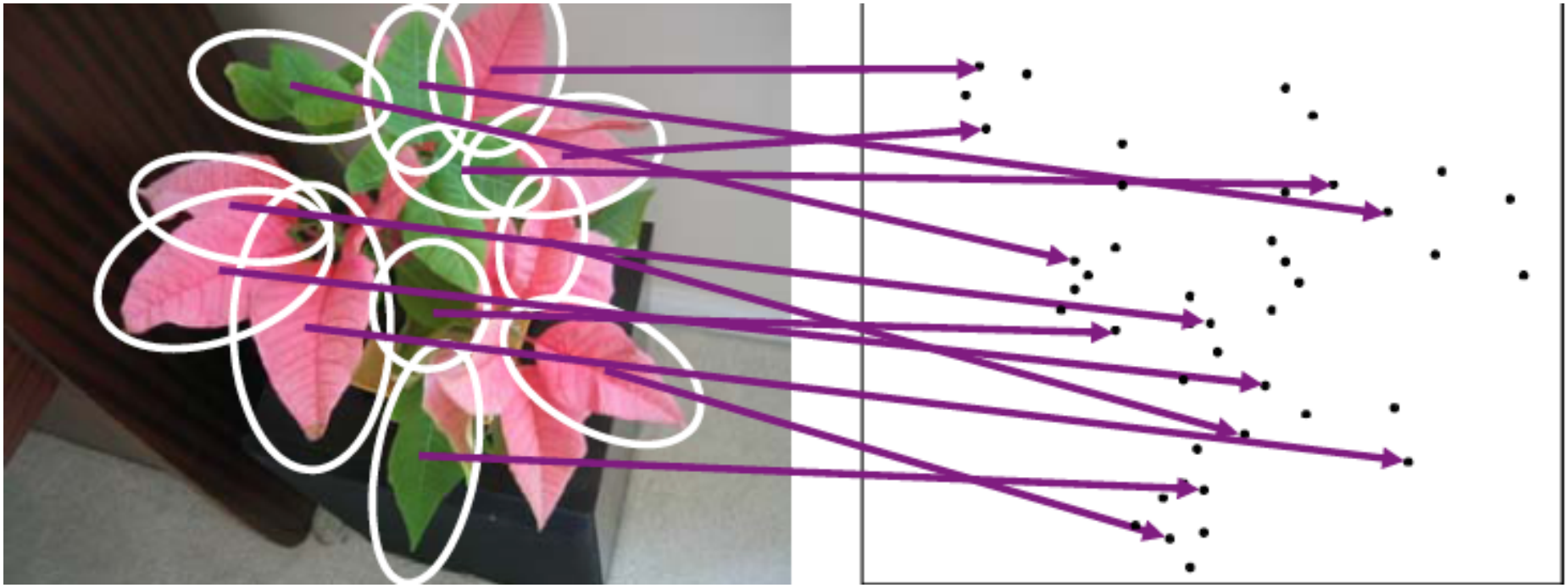
Visual Words: main idea

Extract some local features from a number of images



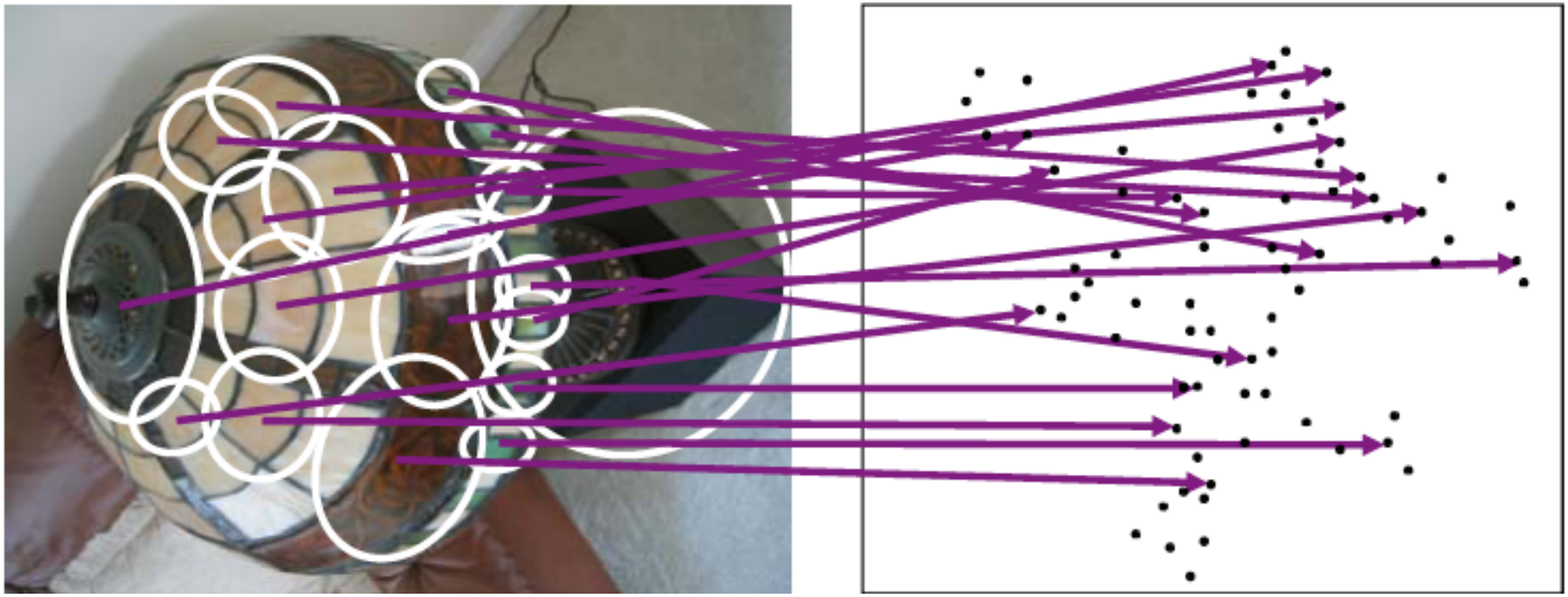
Visual Words: main idea

Extract some local features from a number of images



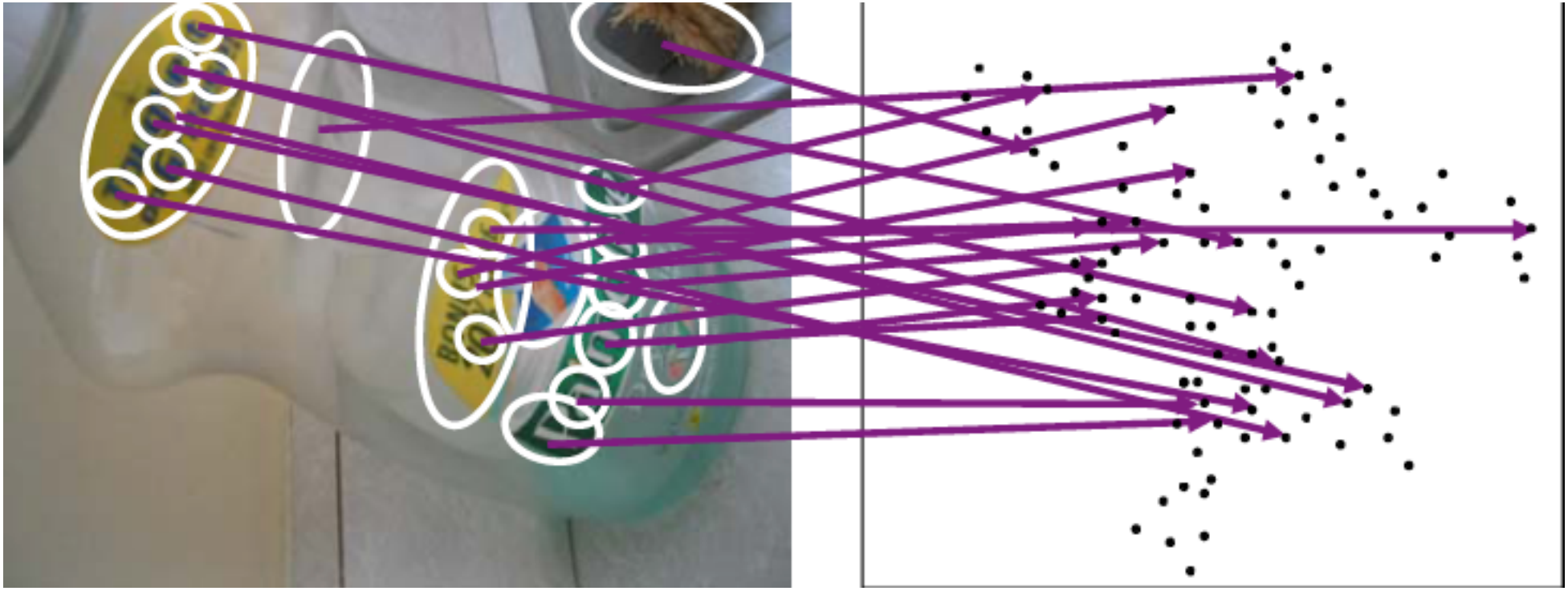
Visual Words: main idea

Extract some local features from a number of images

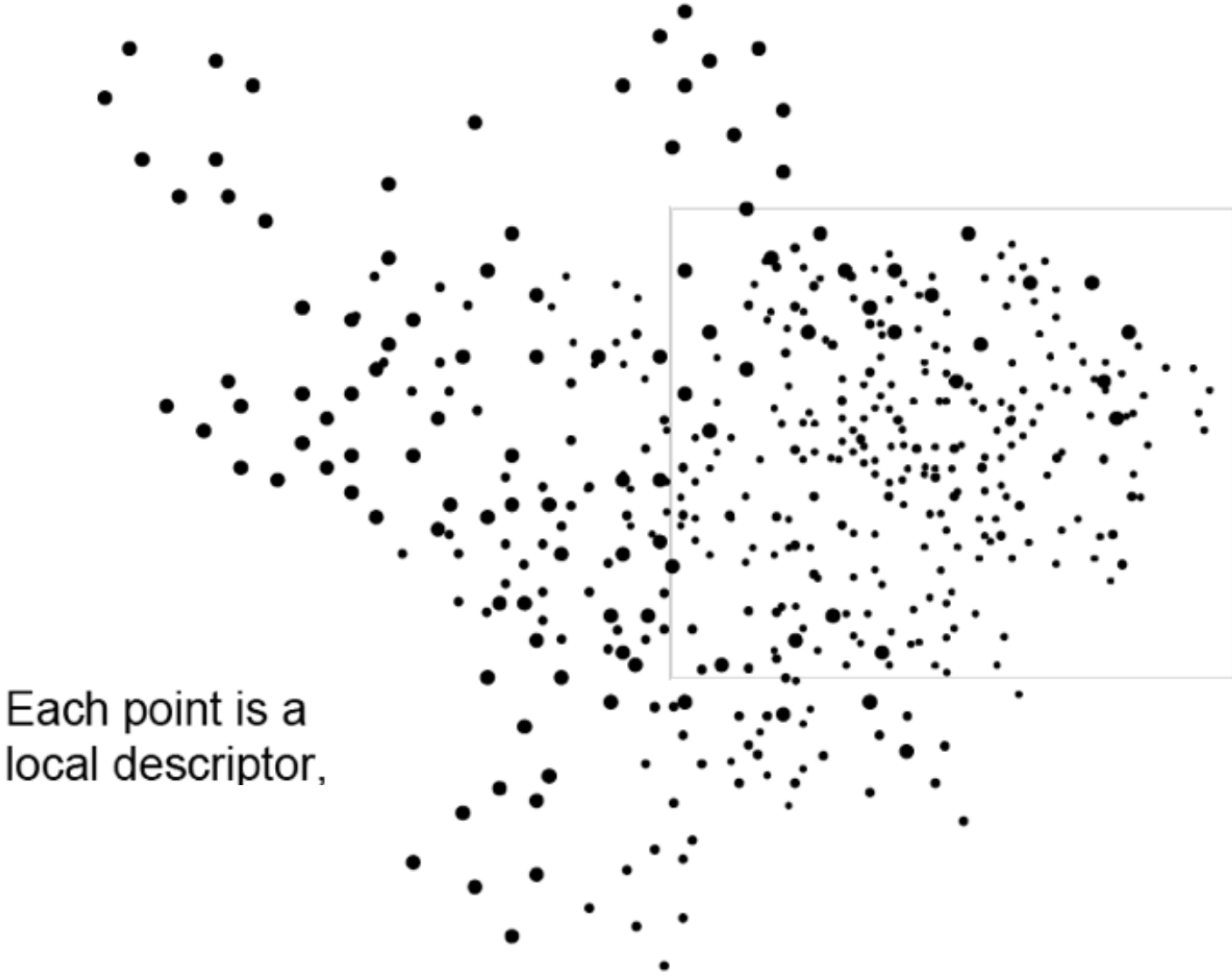


Visual Words: main idea

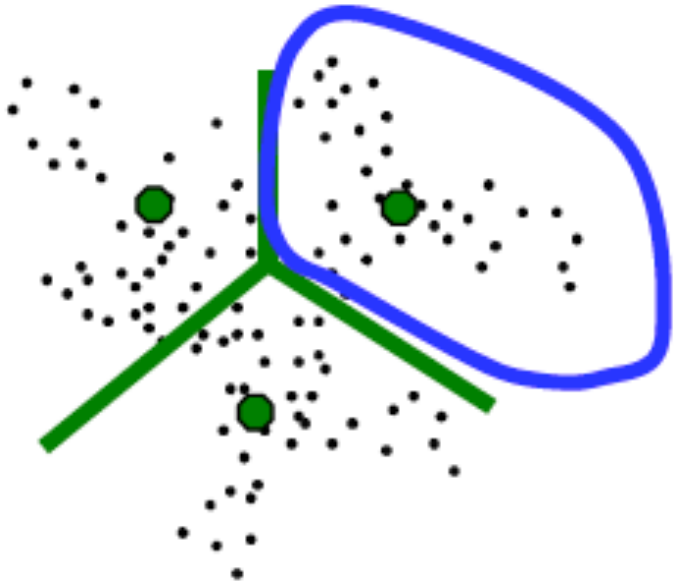
Extract some local features from a number of images



Visual Words: Main Idea



Visual Words



Each group of patches belongs to the same "visual word".

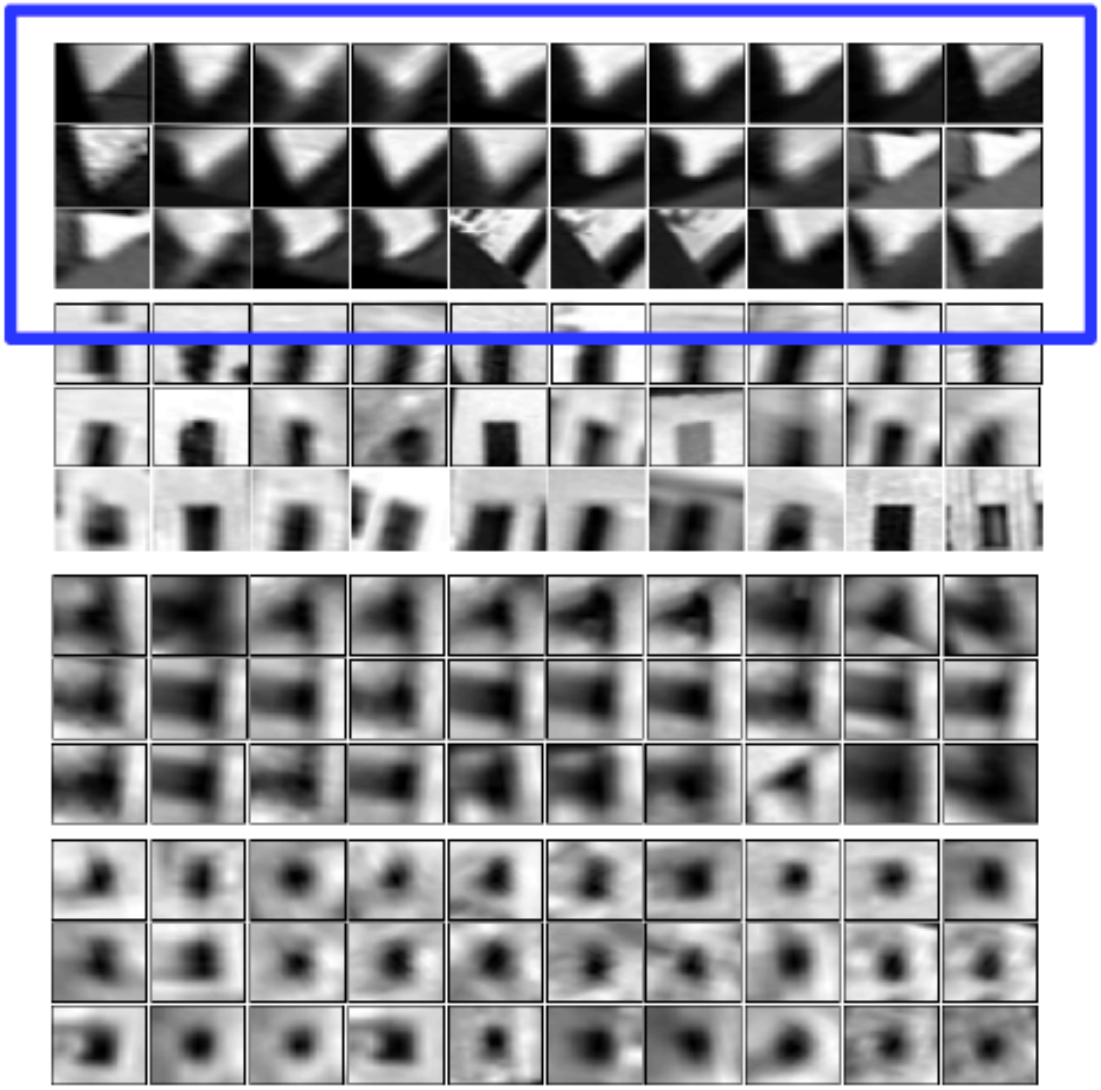
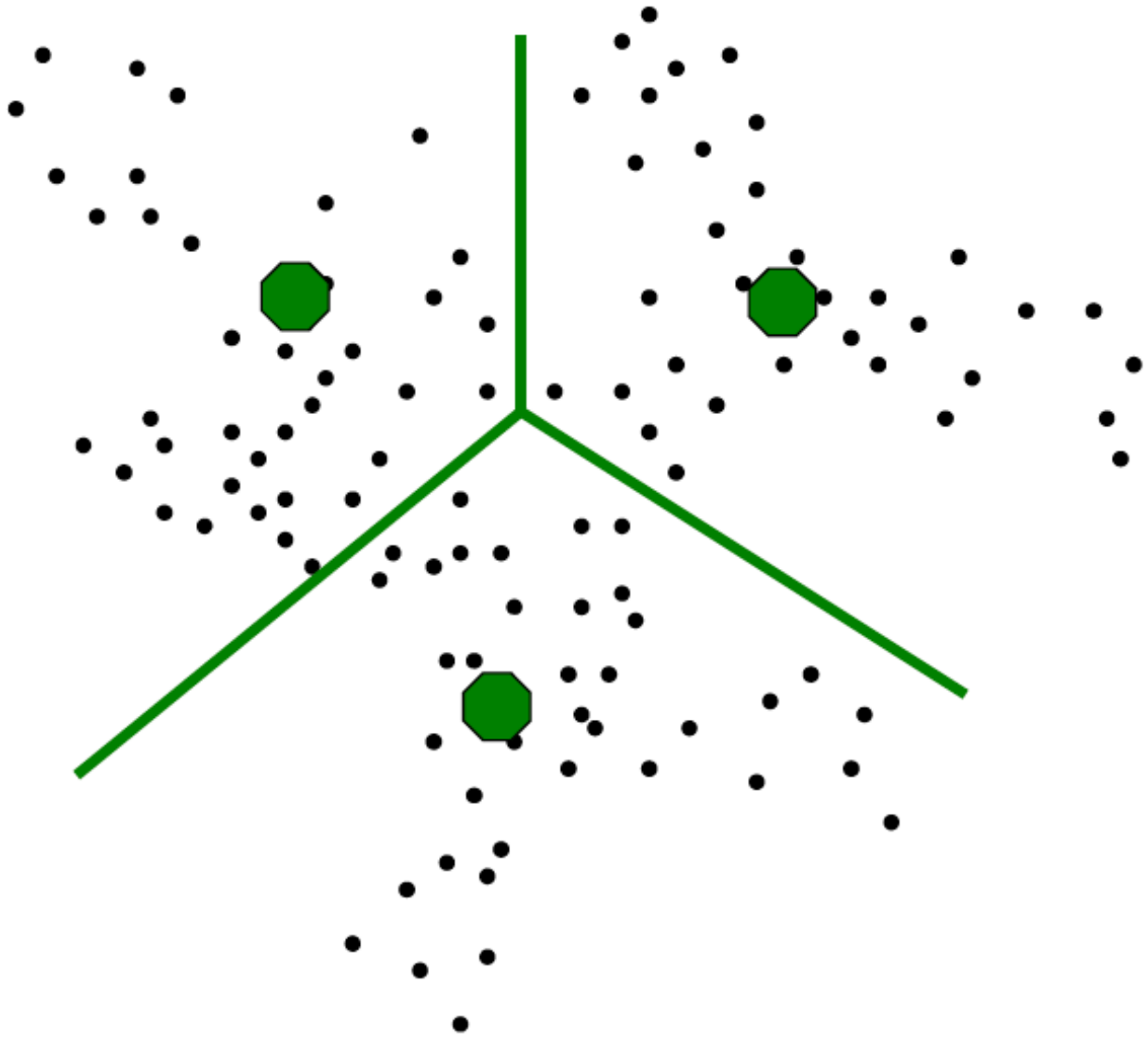


Figure from Sivic & Zisserman, ICCV 2003

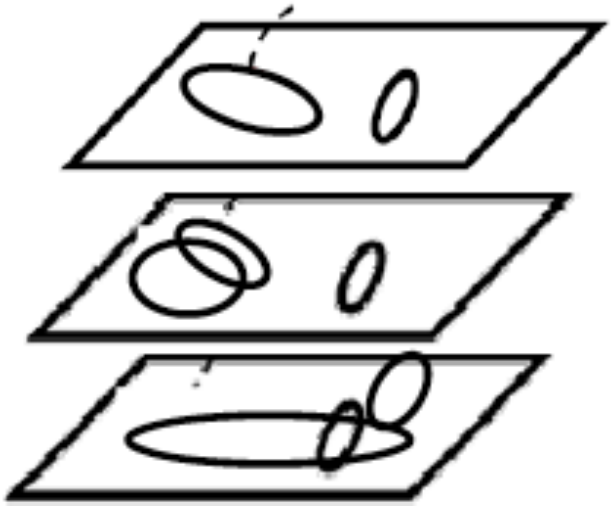
Visual Word Clustering: Vocabulary

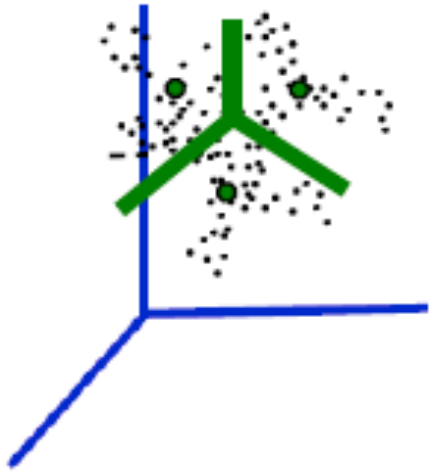


Clusters: Prototype Words

Map high dimensional descriptors to tokens/words by quantizing the feature space:

- Quantize by clustering: each cluster center is a word prototype



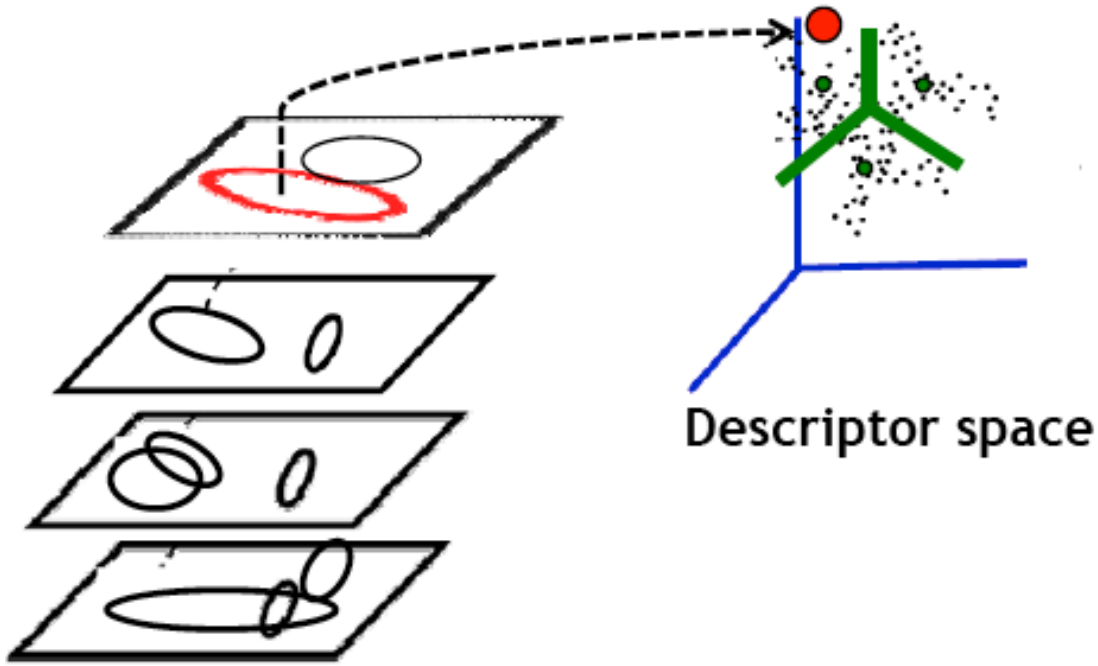


Descriptor space

Clusters: Prototype Words

Map high dimensional descriptors to tokens/words by quantizing the feature space:

- Determine which word to assign to each new image by finding the closest cluster center



Inverted File Index

Database images are loaded into the index mapping words to image numbers

Word #	Image #
1	3
2 …	
7	1, 2
8	3
9	
10 …	
91	2

…

Inverted File Index

New query image is mapped to indices of database images that share a word.

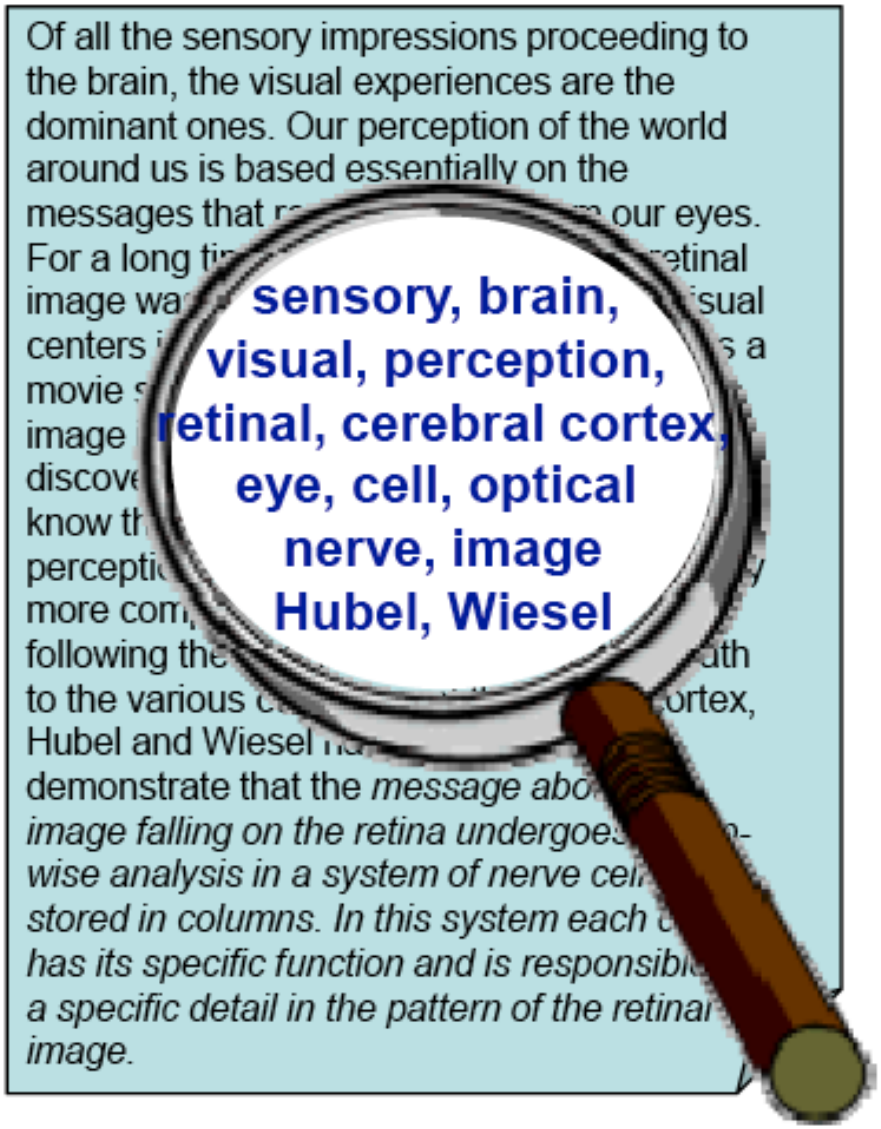
New query image

Word #	Image #
1	3
2	
...	
7	1, 2
8	3
9	
10	
...	
91	2

Visual Documents

If a local region is a “visual word”, how can we summarize an image - i.e. a “visual document”?

Visual Documents



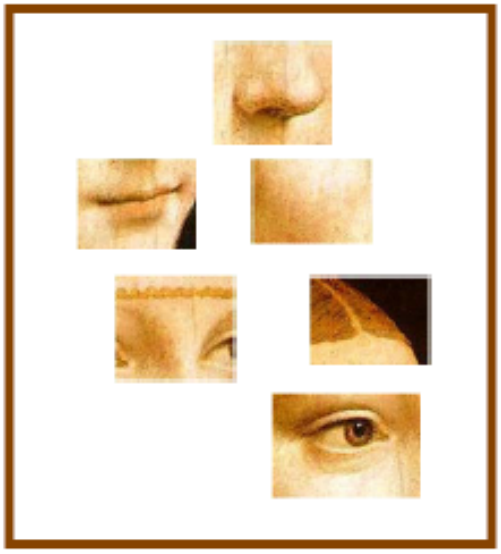
ICCV 2005 short course, L. Fei-Fei

Object → Bag of 'words'

Definition of “BoW”

Independent features

face



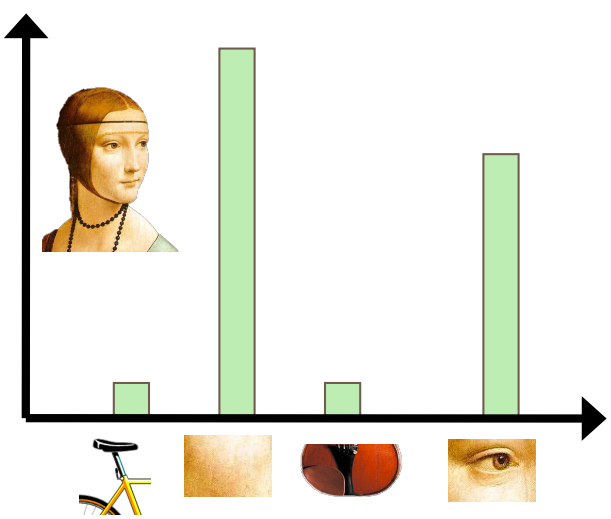
bike

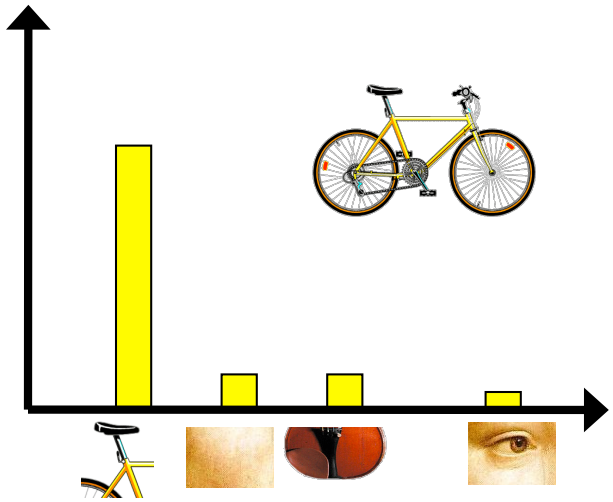
violin

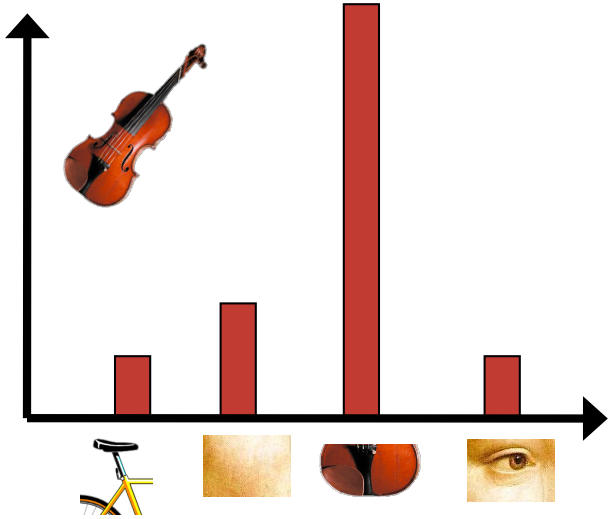
Definition of “BoW”

Independent features

histogram representation

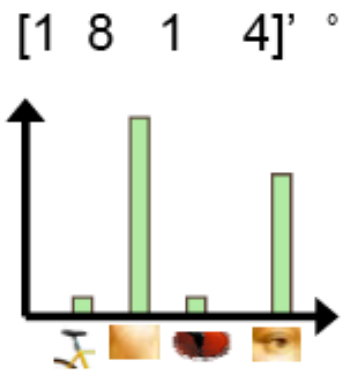


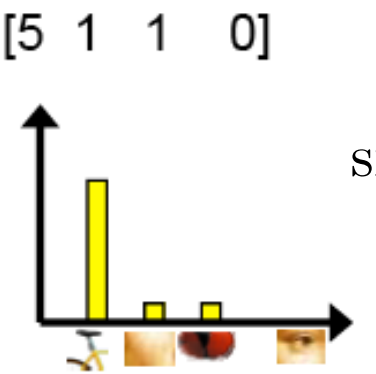




Comparing BoW

Rank frames by normalized scalar product between their (possibly weighted) occurrence counts -- nearest neighbor search for similar images.





$$\text{sim}(d_j, q) = \frac{d.q}{|d||q|} = \frac{\sum_{i=1}^{t} w_{ji} w_{qi}}{\sqrt{\sum_{i=1}^{t} w_{ji}^2} \sqrt{\sum_{i=1}^{t} w_{qi}^2}}$$

$\vec{d}_j$

$\vec{q}$

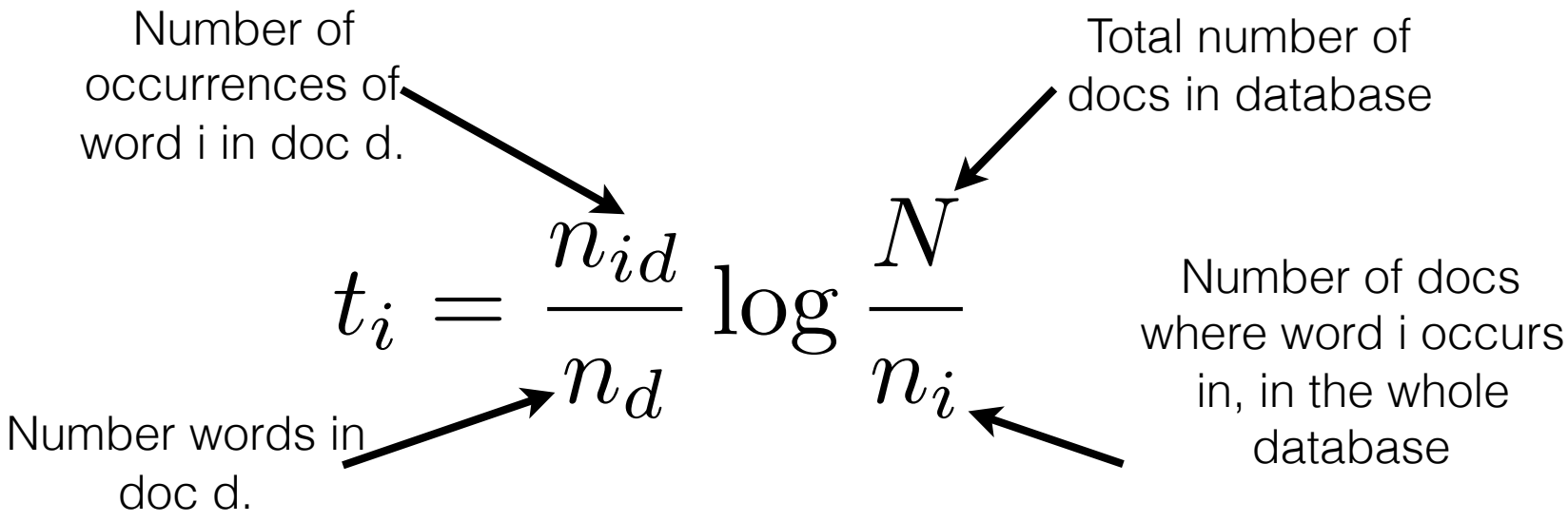
tf-idf weighting

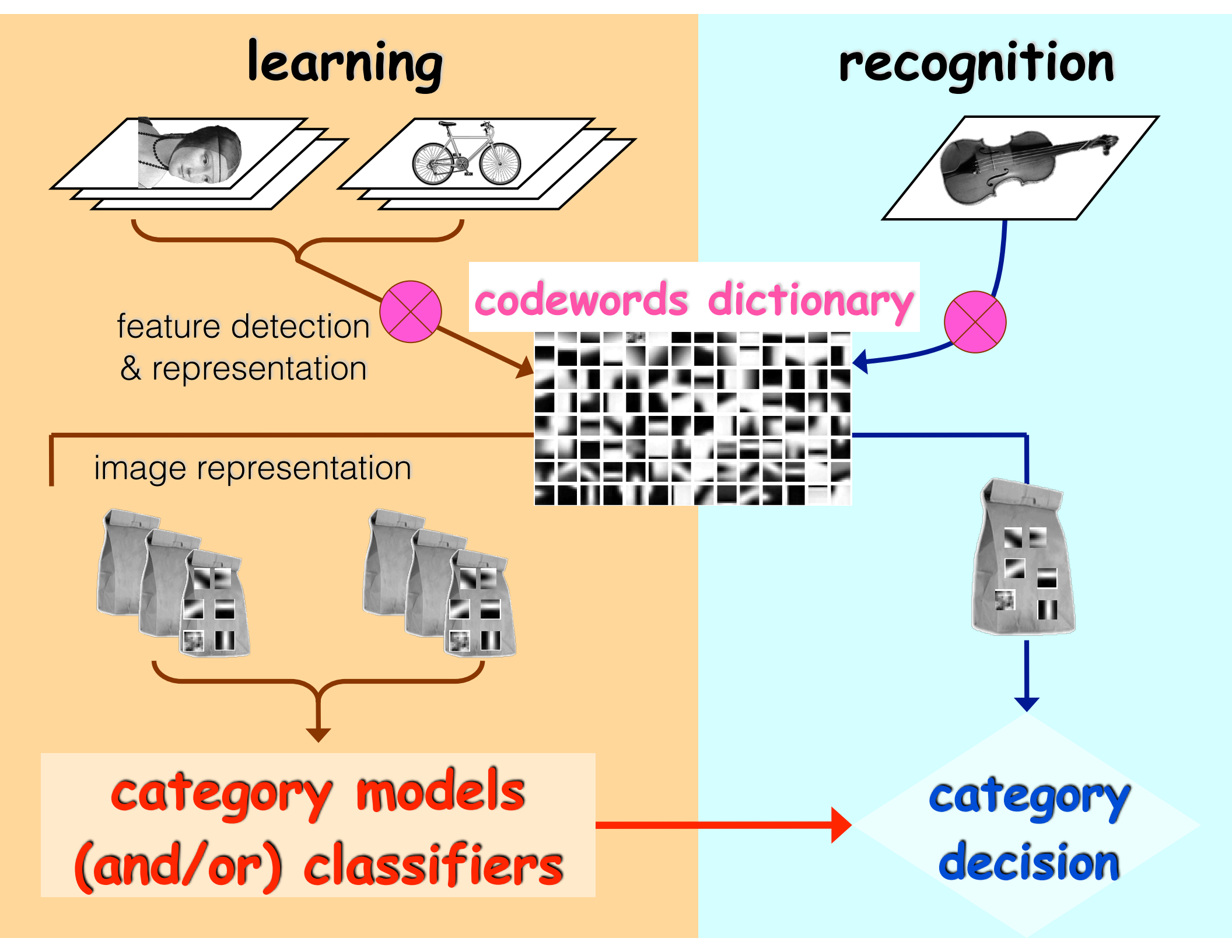
Term **F**requency - **I**nverse **D**ocument **F**requency:

Describe frame by frequency of each word in it

Downweight words that appear often in the database

Standard weighting in text retrieval




learning
recognition
codewords dictionary
feature detection
& representation
image representation
category models
(and/or) classifiers
category
decision

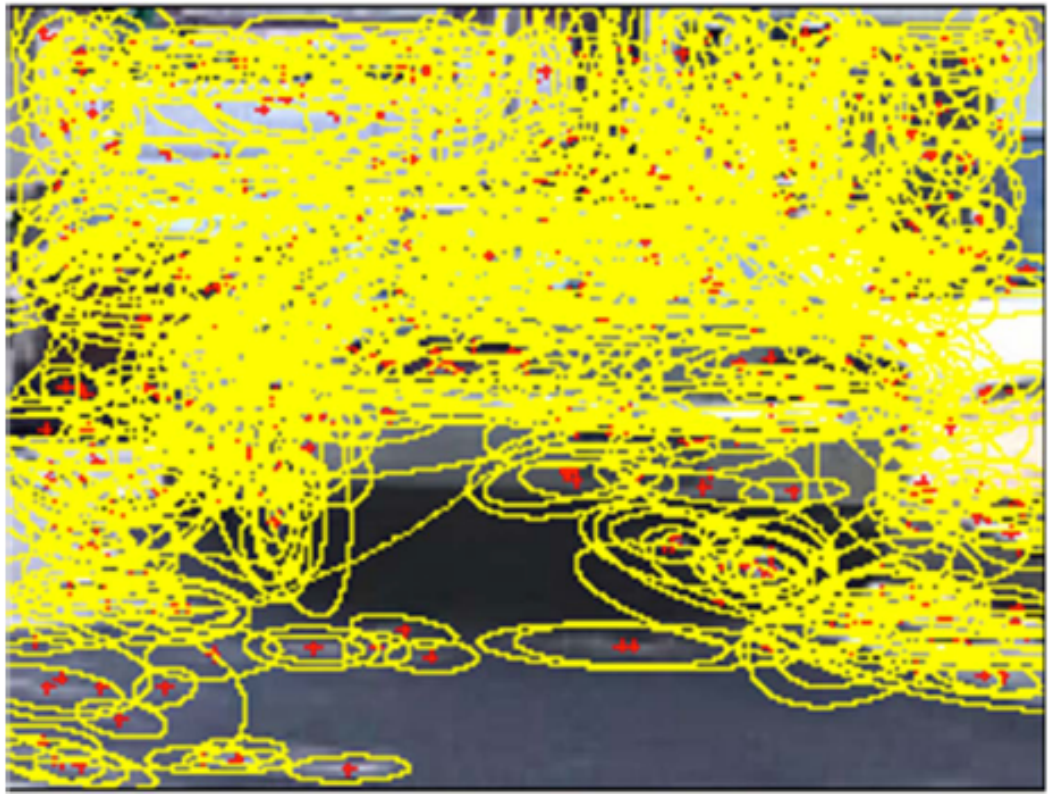
Feature Detection

Regular grid

- Vogel & Schiele, ’03
- Fei-Fei & Perona ’05

Feature Detection

- Regular grid
  - Vogel & Schiele, ’03
  - Fei-Fei & Perona ’05
- Interest Points
  - Csurka et al, ’04
  - Fei-Fei & Perona ’05
  - Sivic et al ’05



Matching with Features

Problem 1:

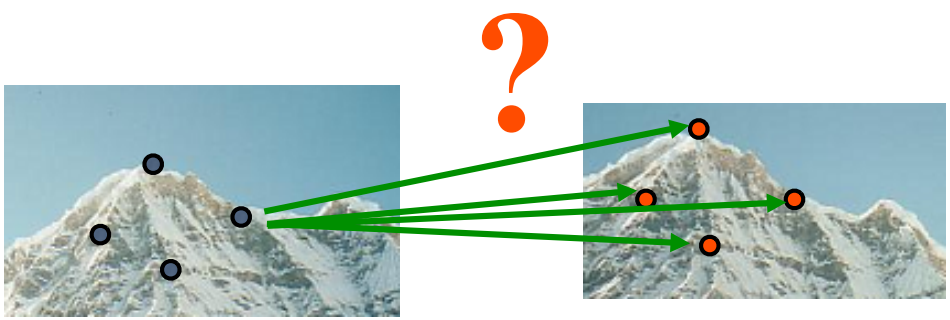
Detect the *same* points in both images

Feature detection must be repeatable

Matching with Features

Problem 2:

Match features correctly

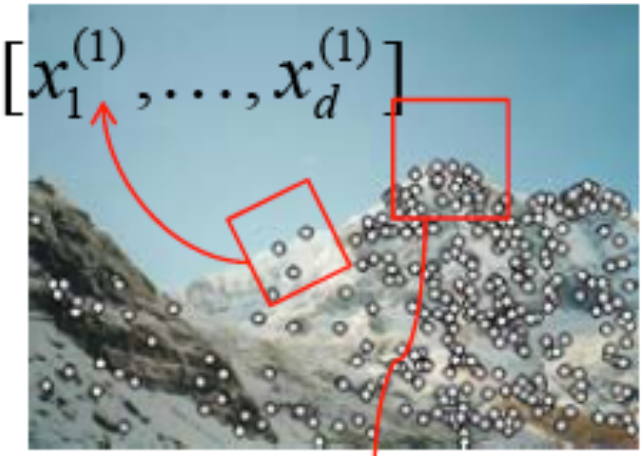


We need a good description of the features

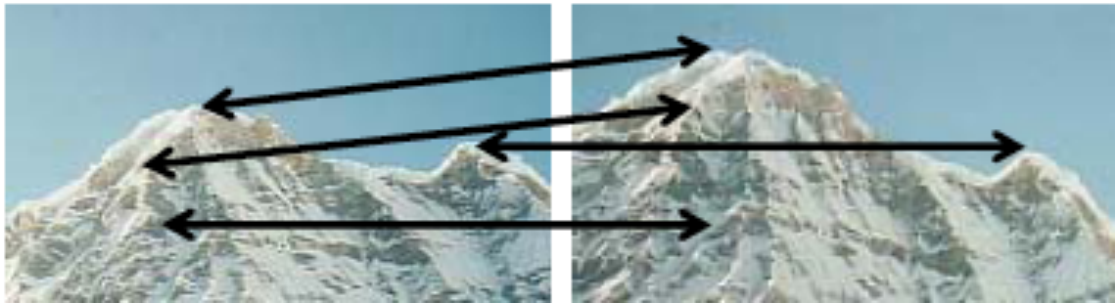
Using Local Features

1. Detection: Identify interesting points
2. Description: Extract vectors describing the local features around the interesting points
3. Matching: Determine correspondence between descriptors in two views

$$\mathbf{x}_1 = [x_1^{(1)}, \ldots, x_d^{(1)}]$$



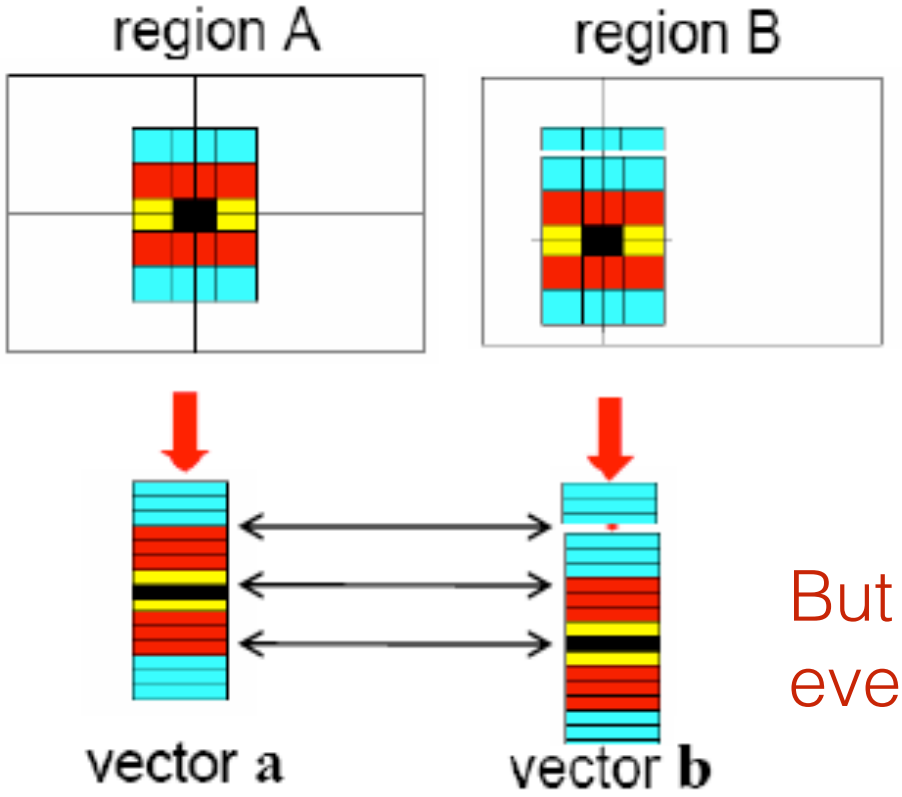
$$\mathbf{x}_2 = [x_1^{(2)}, \ldots, x_d^{(2)}]$$



Local Features: Desired Properties

- Repeatability
  - The same feature can be found in several images, despite geometric and photometric transformations
- Saliency
  - Each feature has a distinctive description
- Compactness and Efficiency
  - Lot less features than pixels
- Locality
  - A feature occupies a relatively small area
  - Robust to occlusion and clutter

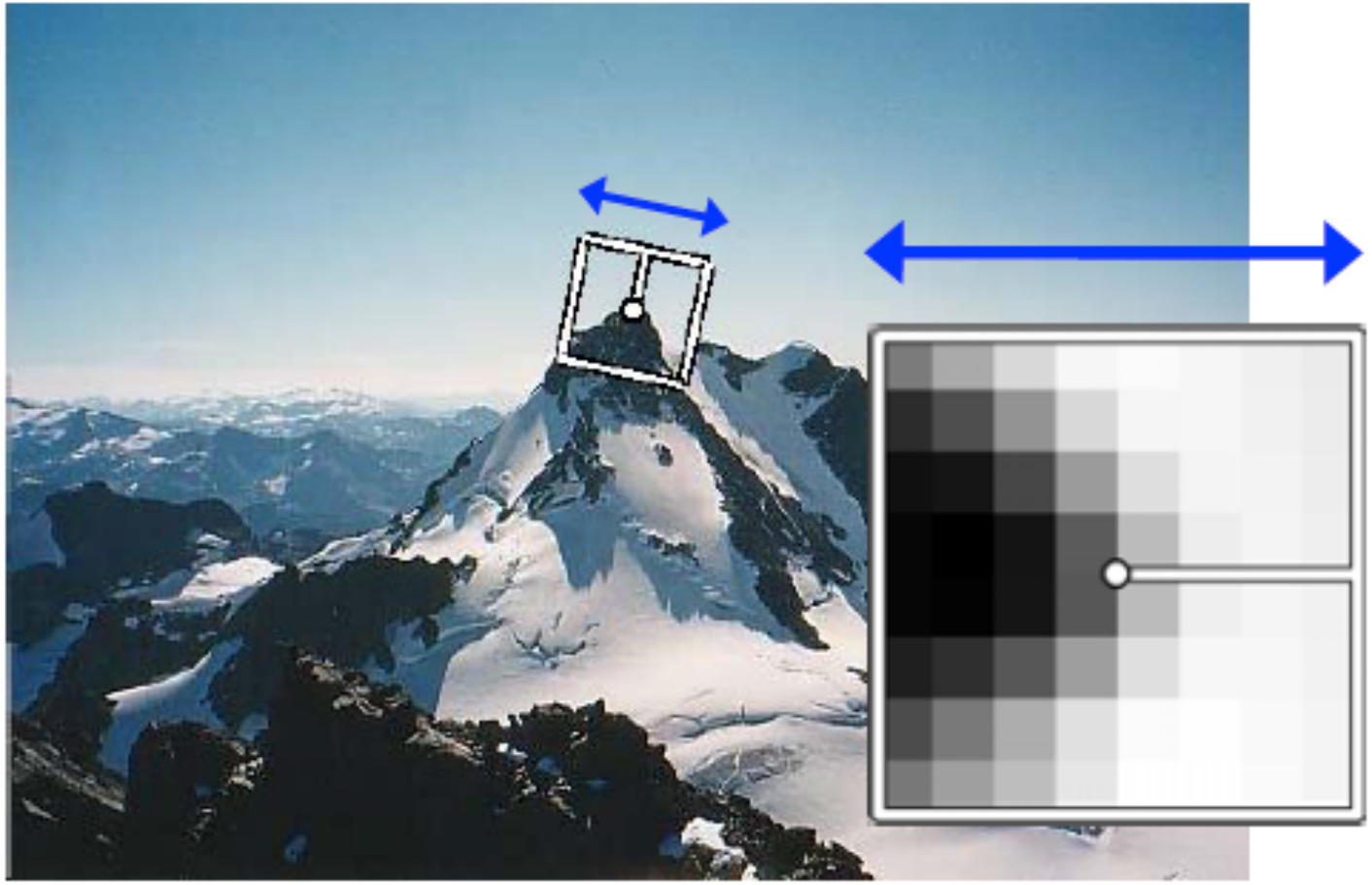
Raw Patches as Local Descriptors



The simplest way to describe the neighborhood around an interest point is to write down the list of intensities to form a feature vector.

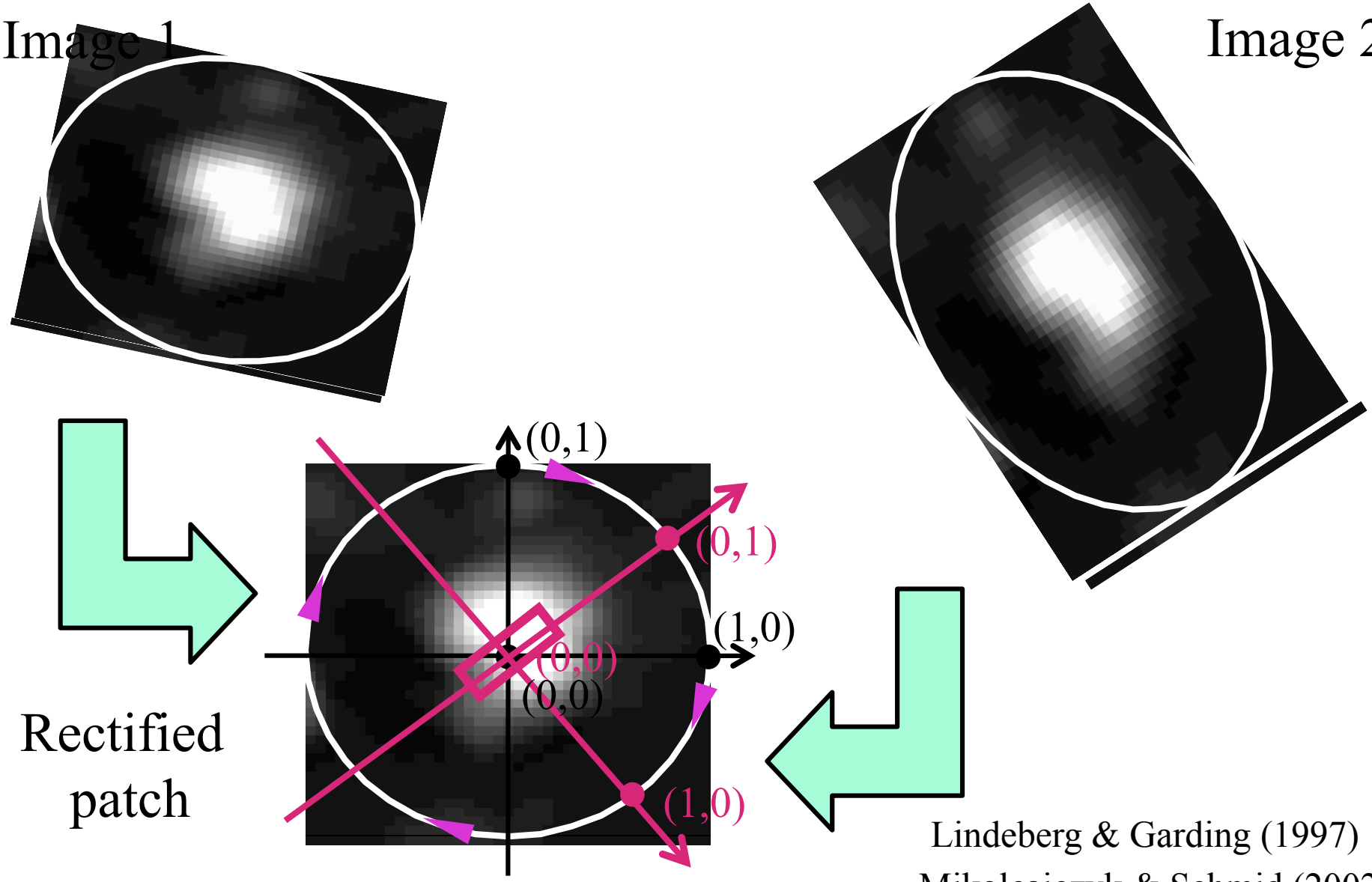
But this is very sensitive to even small shifts an rotations

Rotation Invariance



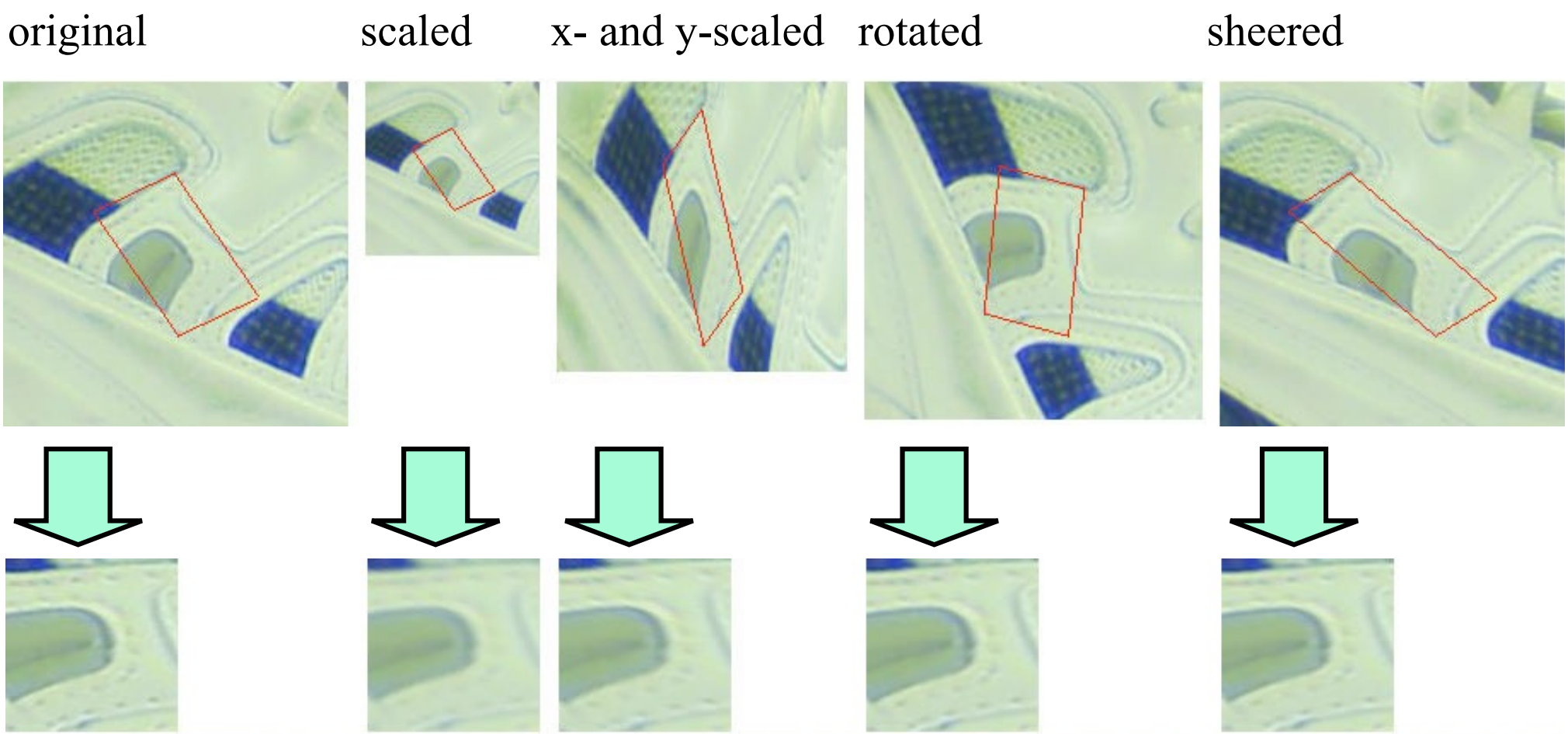
Rotate patch according to its dominant gradient orientation:
“canonical” representation.

Affine adaptation/Rectification process



Lindeberg & Garding (1997)
Mikolcajczyk & Schmid (2002)

Rectification Example

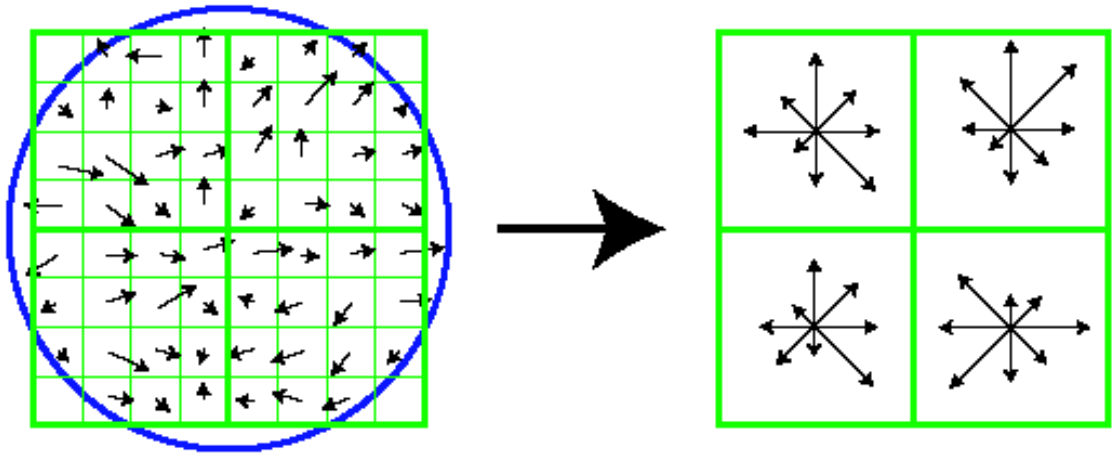


Descriptor overview:

Determine scale (by maximizing DoG in scale and in space),
local orientation as the dominant gradient direction.
Use this scale and orientation to make all further computations invariant to scale and rotation.

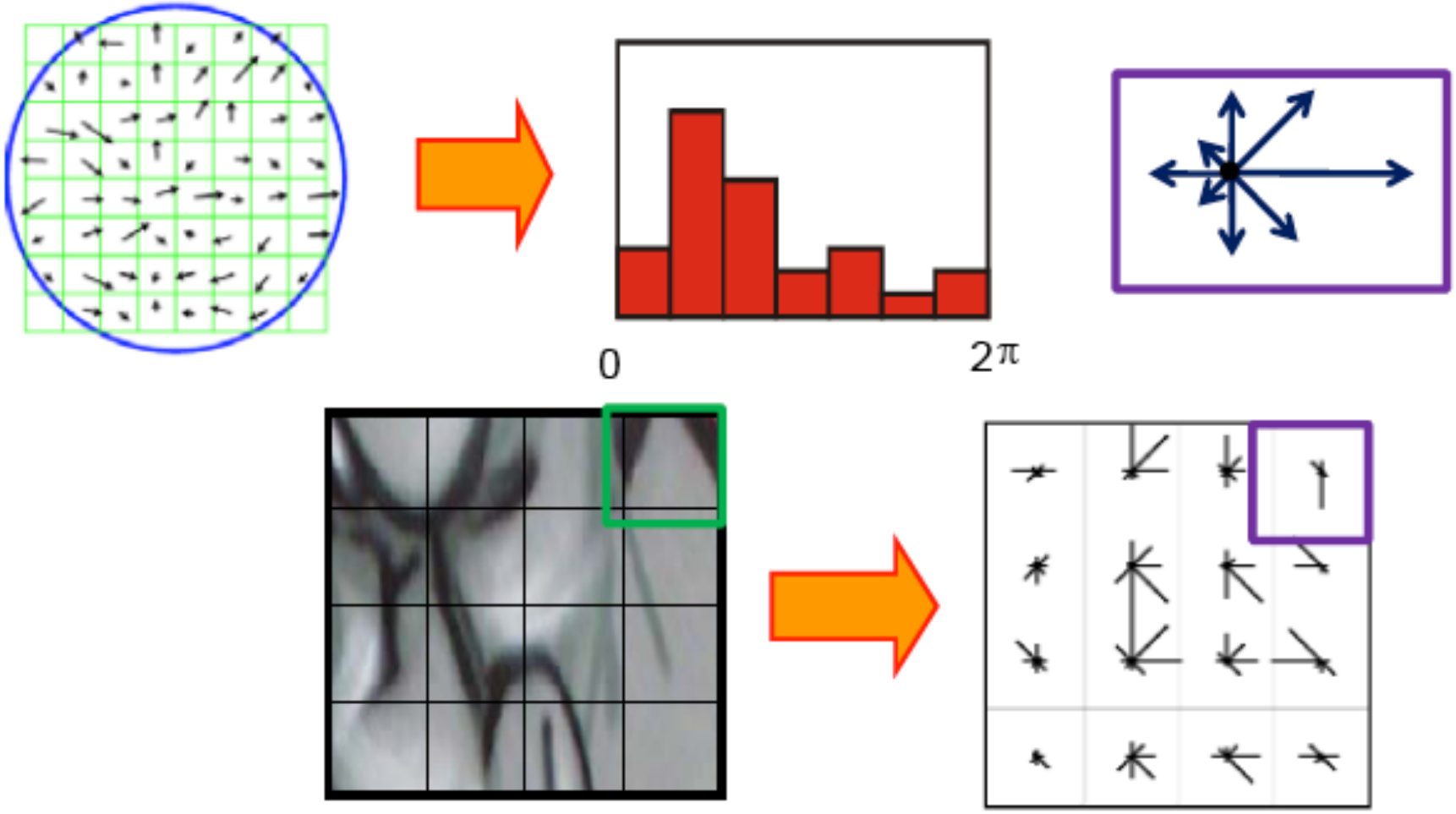
Compute gradient orientation histograms of several small windows (128 values for each point)

Normalize the descriptor to make it invariant to intensity change



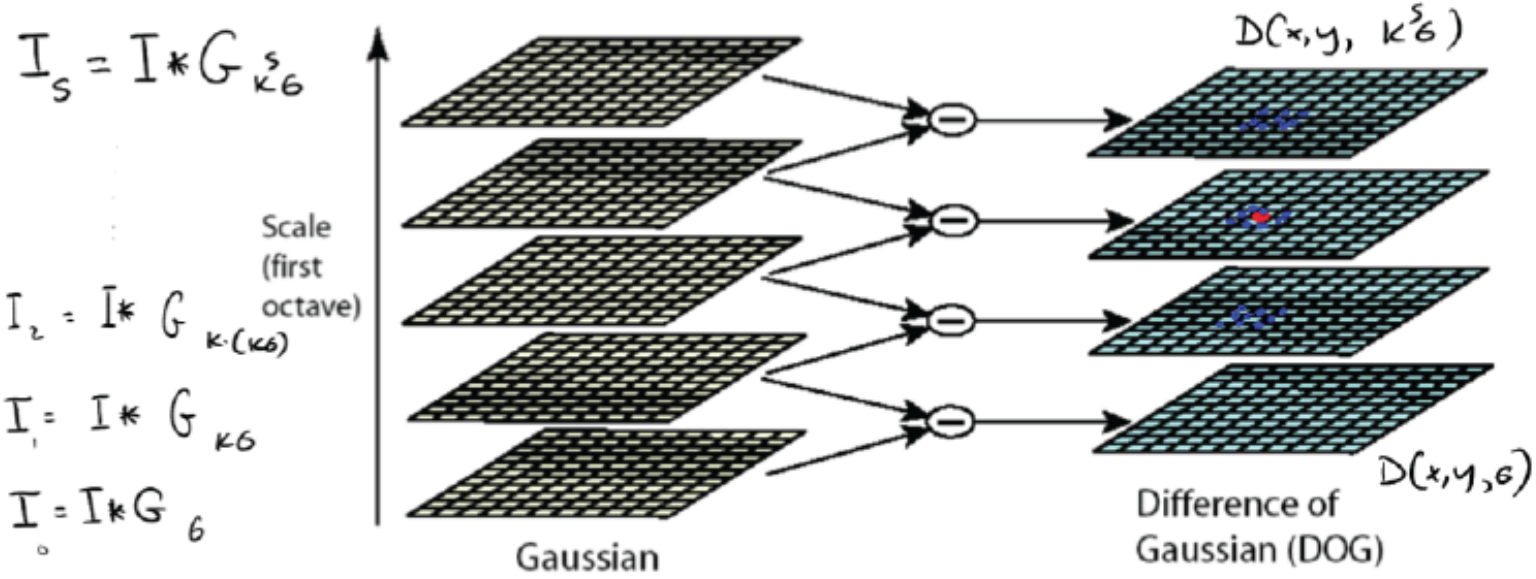
SIFT DESCRIPTOR (Lowe 2004)

Use histograms to bin pixels within sub-patches according to their orientation.



SIFT Descriptor

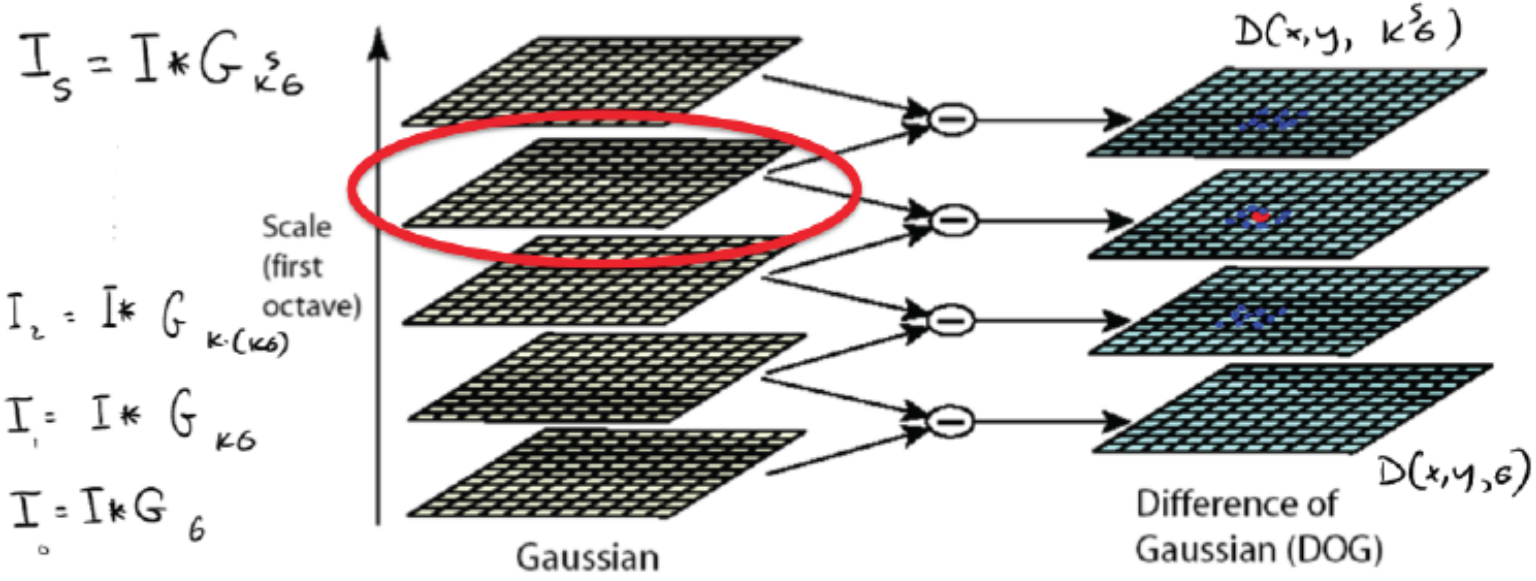
1. Our scale invariant interest point detector gives scale ρ for each keypoint



[Adopted from: F. Flores-Mangas]

SIFT Descriptor

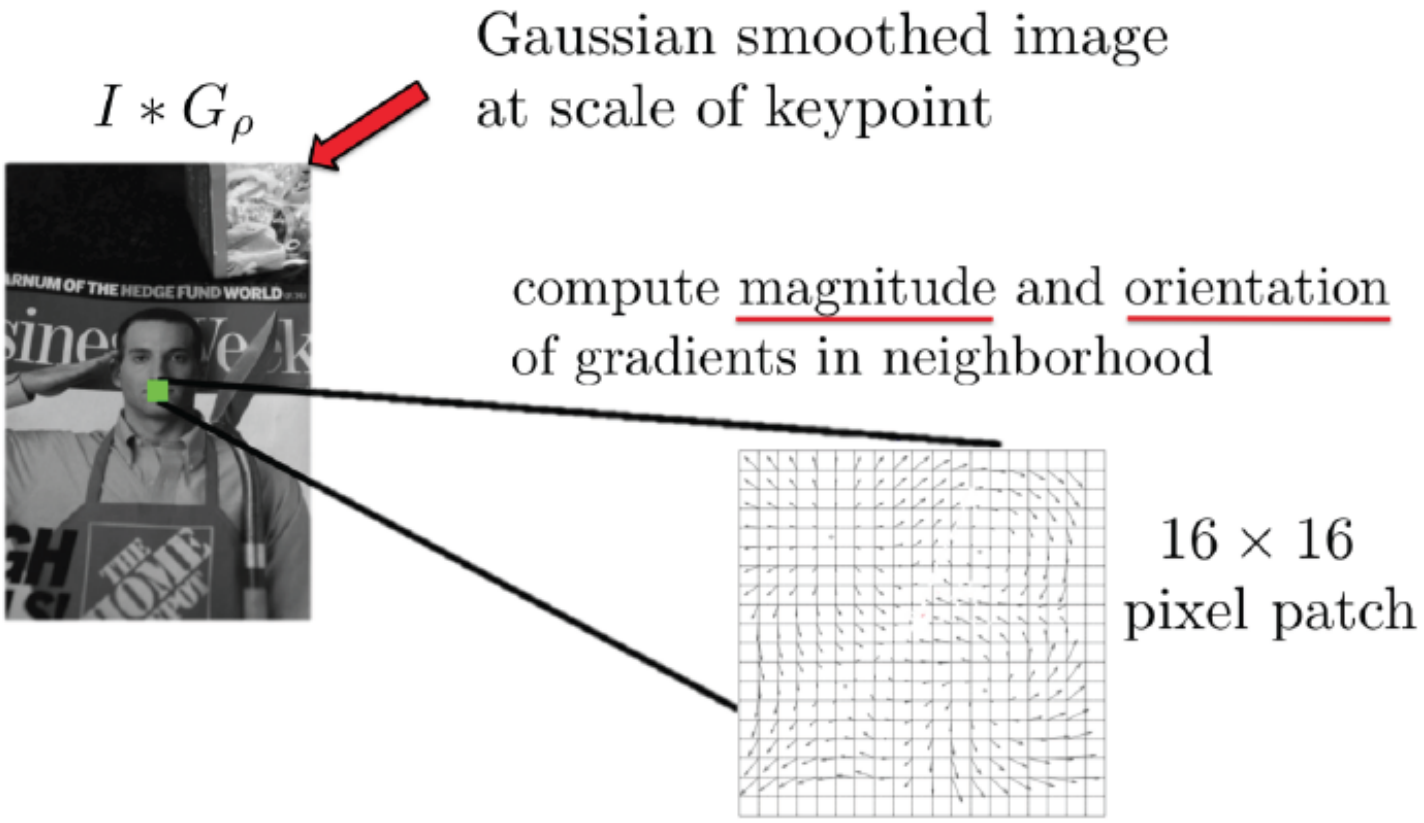
2. For each keypoint, we take the Gaussian-blurred image at corresponding scale ρ



[Adopted from: F. Flores-Mangas]

SIFT Descriptor

3. Compute the gradient magnitude and orientation in neighborhood of each keypoint



[Adopted from: F. Flores-Mangas]

SIFT Descriptor

3. Compute the gradient magnitude and orientation in neighborhood of each keypoint

magnitude of gradient:

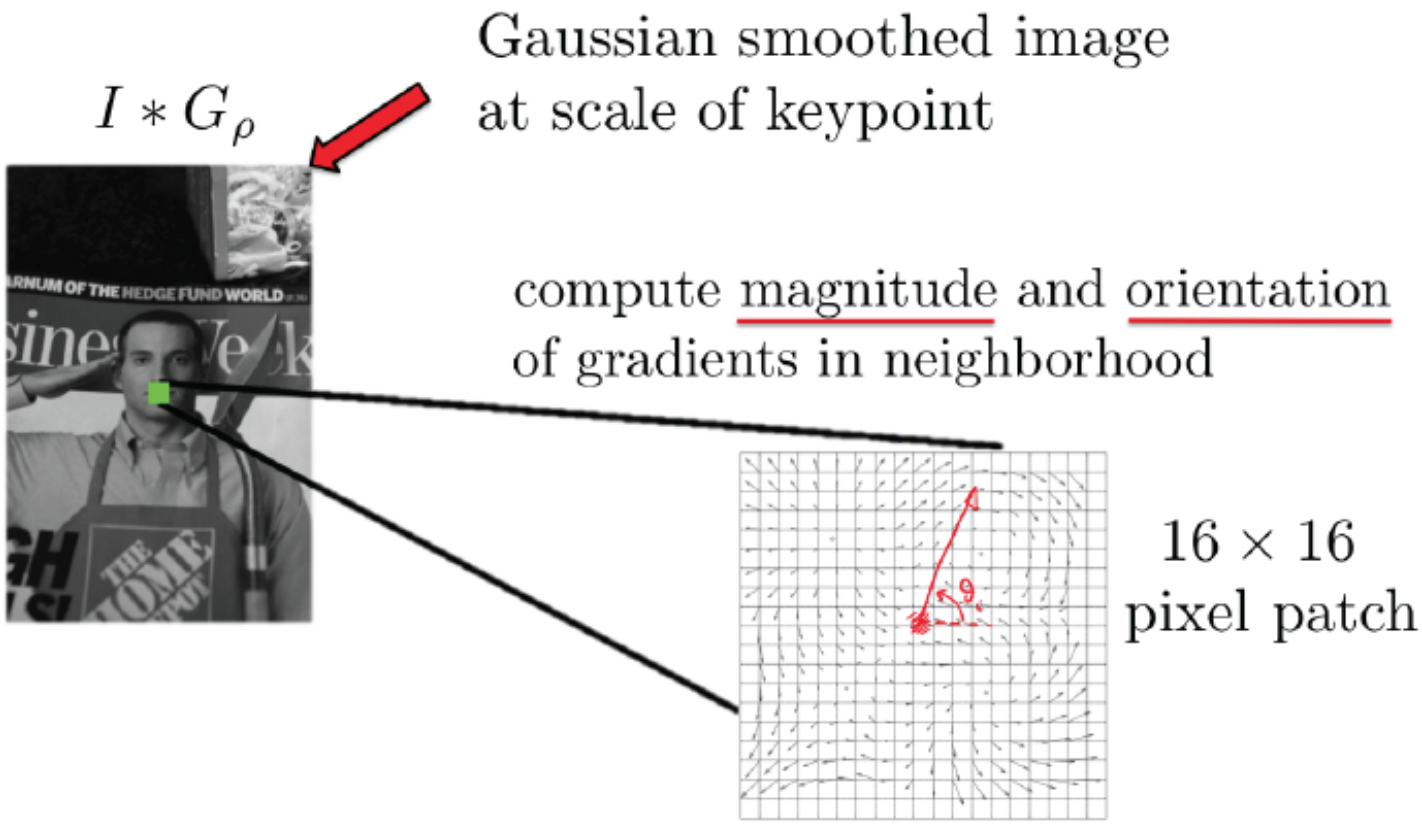
$$|\nabla I(x,y)| = \sqrt{\left(\frac{\partial(I(x,y) * G_\rho)}{\partial x}\right)^2 + \left(\frac{\partial(I(x,y) * G_\rho)}{\partial y}\right)^2}$$

gradient orientation:

$$\theta(x,y) = \arctan\left(\frac{\partial I * G_\rho}{\partial y} / \frac{\partial I * G_\rho}{\partial x}\right)$$

SIFT Descriptor

4. Compute dominant orientation of each keypoint. How?

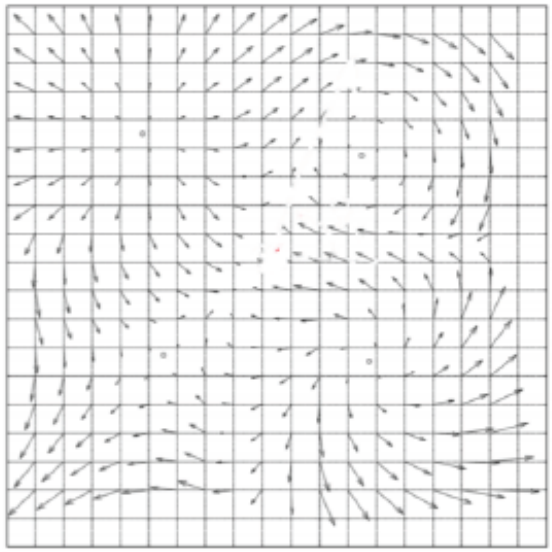


[Adopted from: F. Flores-Mangas]

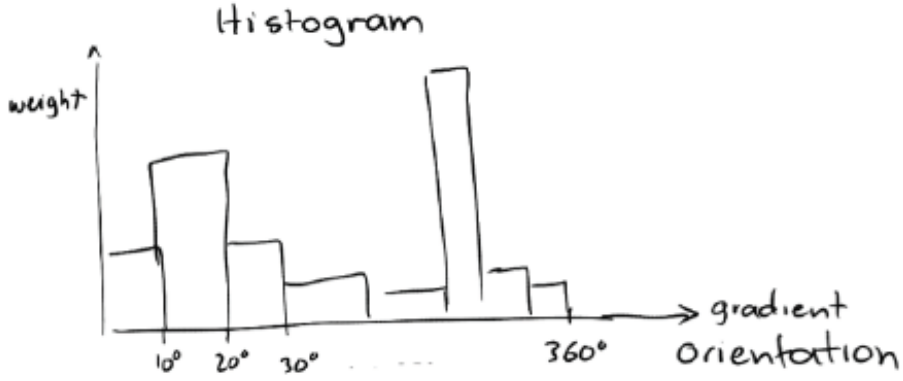
SIFT Descriptor

- Compute a histogram of gradient orientations, each bin covers 10°

16×16



compute histograms of orientations by orientation increments of 10°

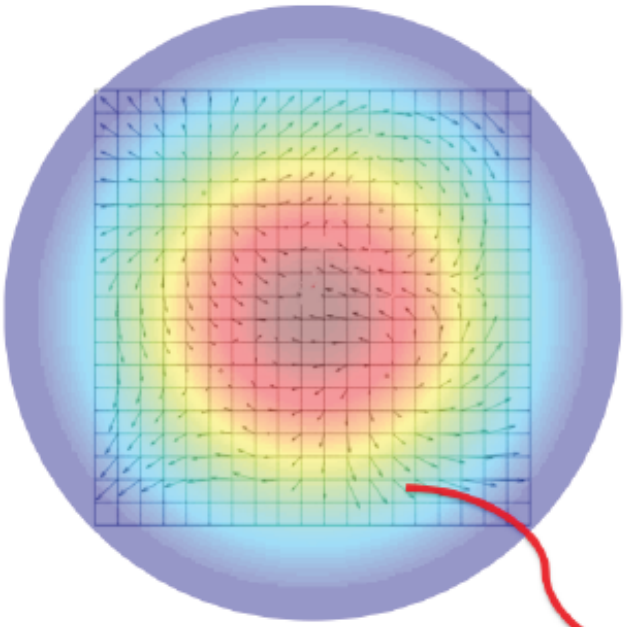


[Adopted from: F. Flores-Mangas]

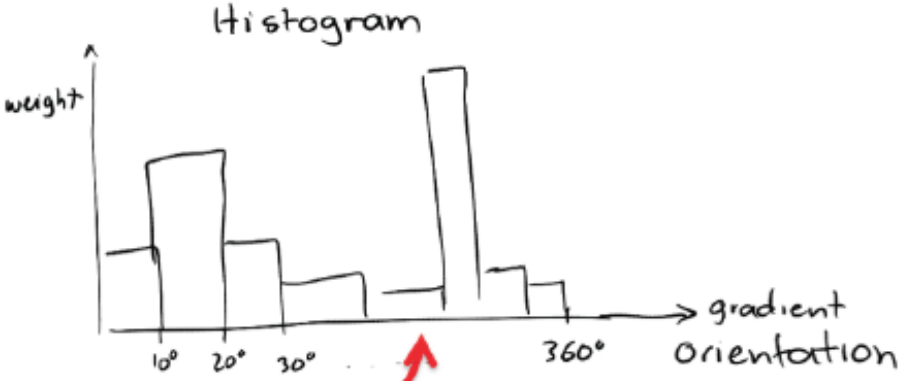
SIFT Descriptor: Dominant Orientation

- Compute a histogram of gradient orientations, each bin covers 10°
- Orientations closer to the keypoint center should contribute more

$G_{1.5\rho}$



weight influence of orientation based on distance from center

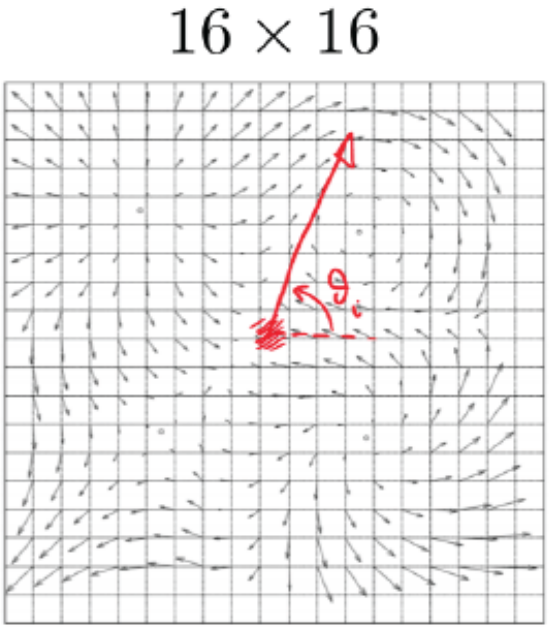


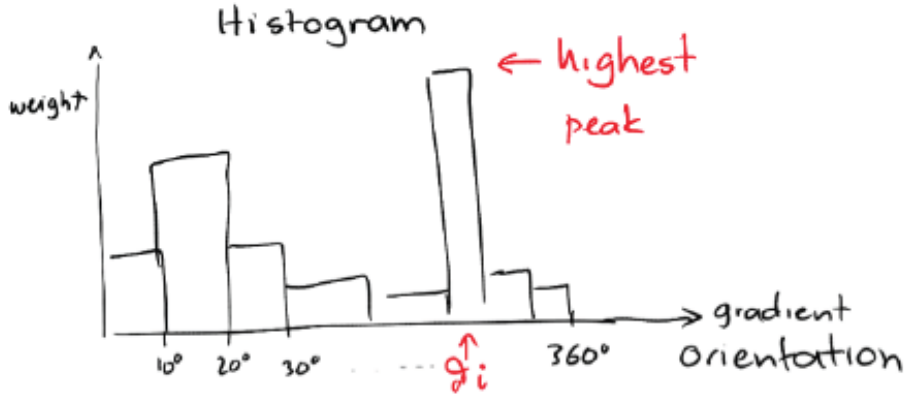
$$|\nabla I(x,y)| \cdot G_{1.5\rho}(d)$$

[Adopted from: F. Flores-Mangas]

SIFT Descriptor: Dominant Orientation

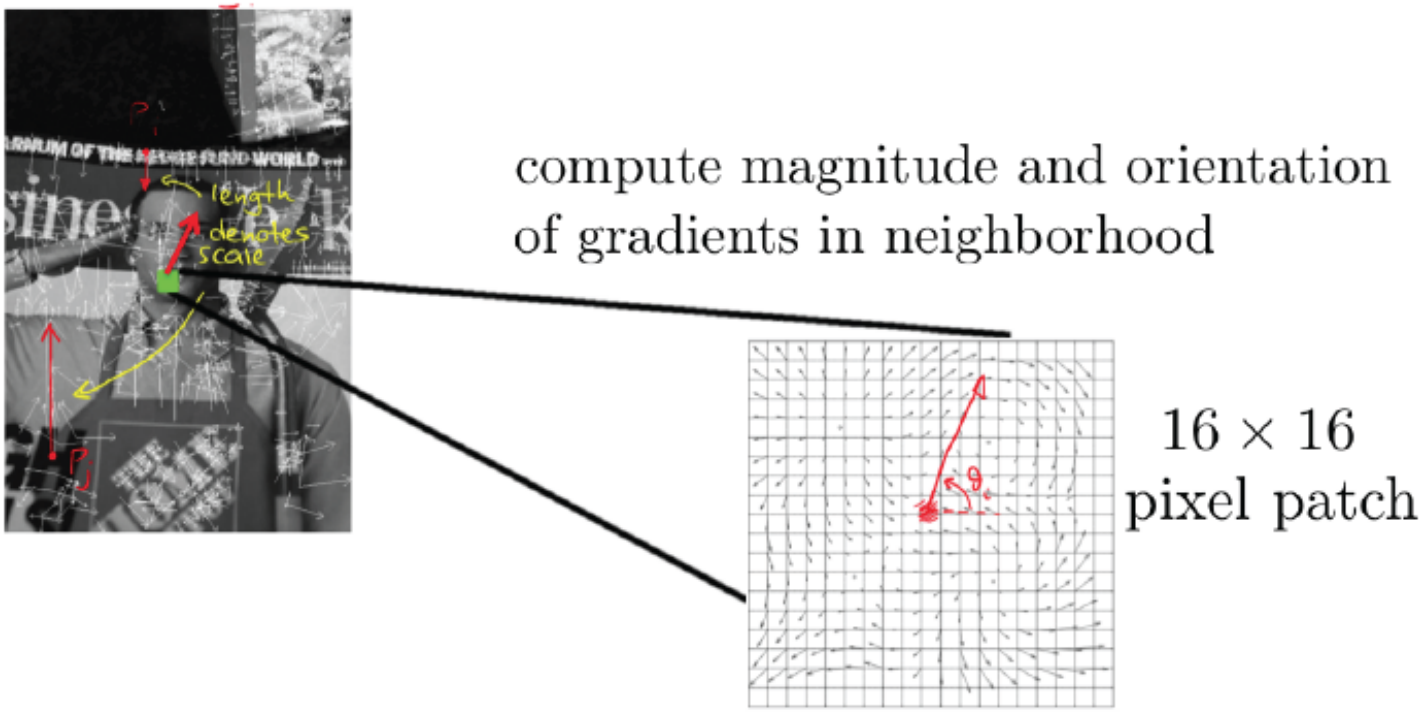
- Compute a histogram of gradient orientations, each bin covers 10°
- Orientations closer to the keypoint center should contribute more
- Orientation giving the peak in the histogram is the keypoint's orientation





SIFT Descriptor

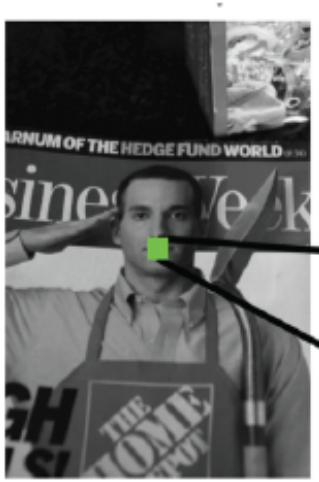
4. Compute dominant orientation



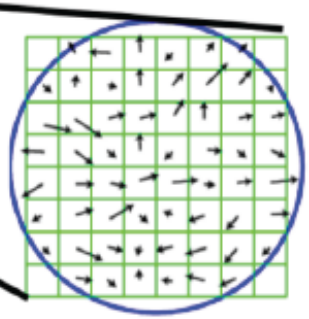
[Adopted from: F. Flores-Mangas]

SIFT Descriptor

5. Compute a 128 dimensional descriptor: 4×4 grid, each cell is a histogram of 8 orientation bins relative to dominant orientation



compute descriptor, relative to dominant orientation



128 dim descriptor

each descriptor has:

$P_i = (x_i, y_i, \rho_i, \vartheta_i)$ and f_i ... 128 dim vector

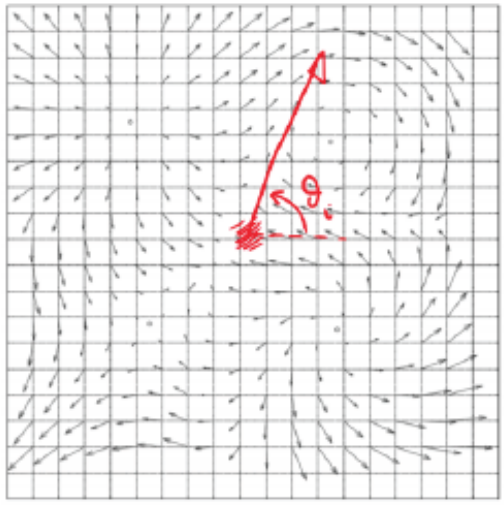
location scale orientation feature vector

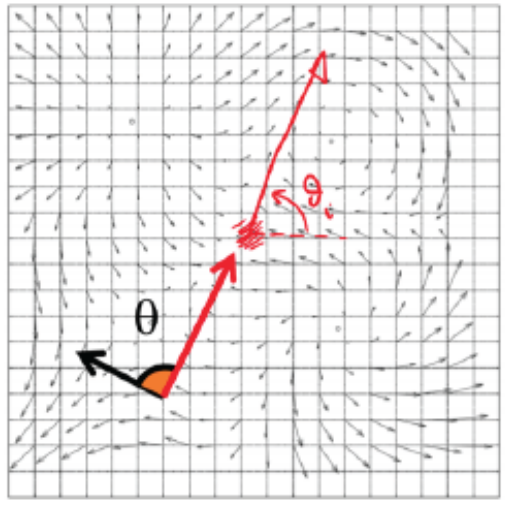
[Adopted from: F. Flores-Mangas]

SIFT Descriptor: Computing the Feature Vector

- Compute the orientations **relative** to the **dominant orientation**

16×16 patch
centered in (x_i, y_i)



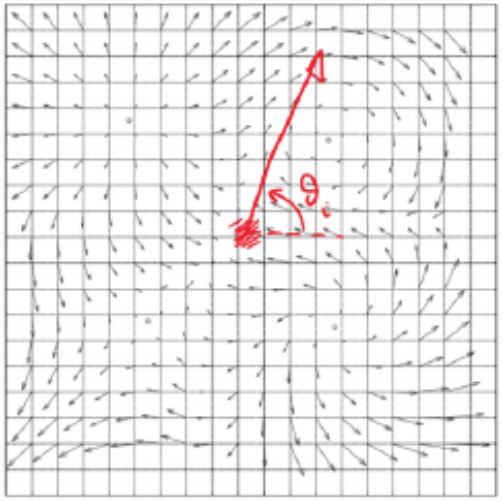


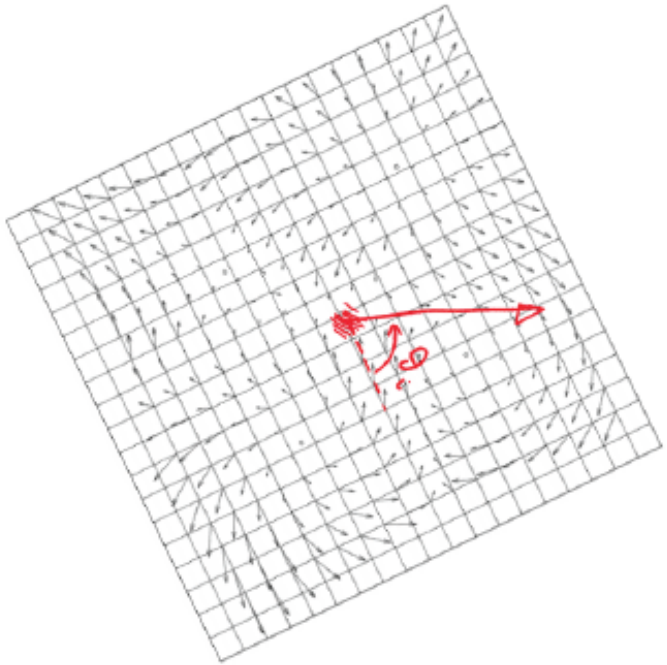
[Adopted from: F. Flores-Mangas]

SIFT Descriptor: Computing the Feature Vector

- Compute the orientations **relative** to the **dominant orientation**

16×16 patch
centered in (x_i, y_i)

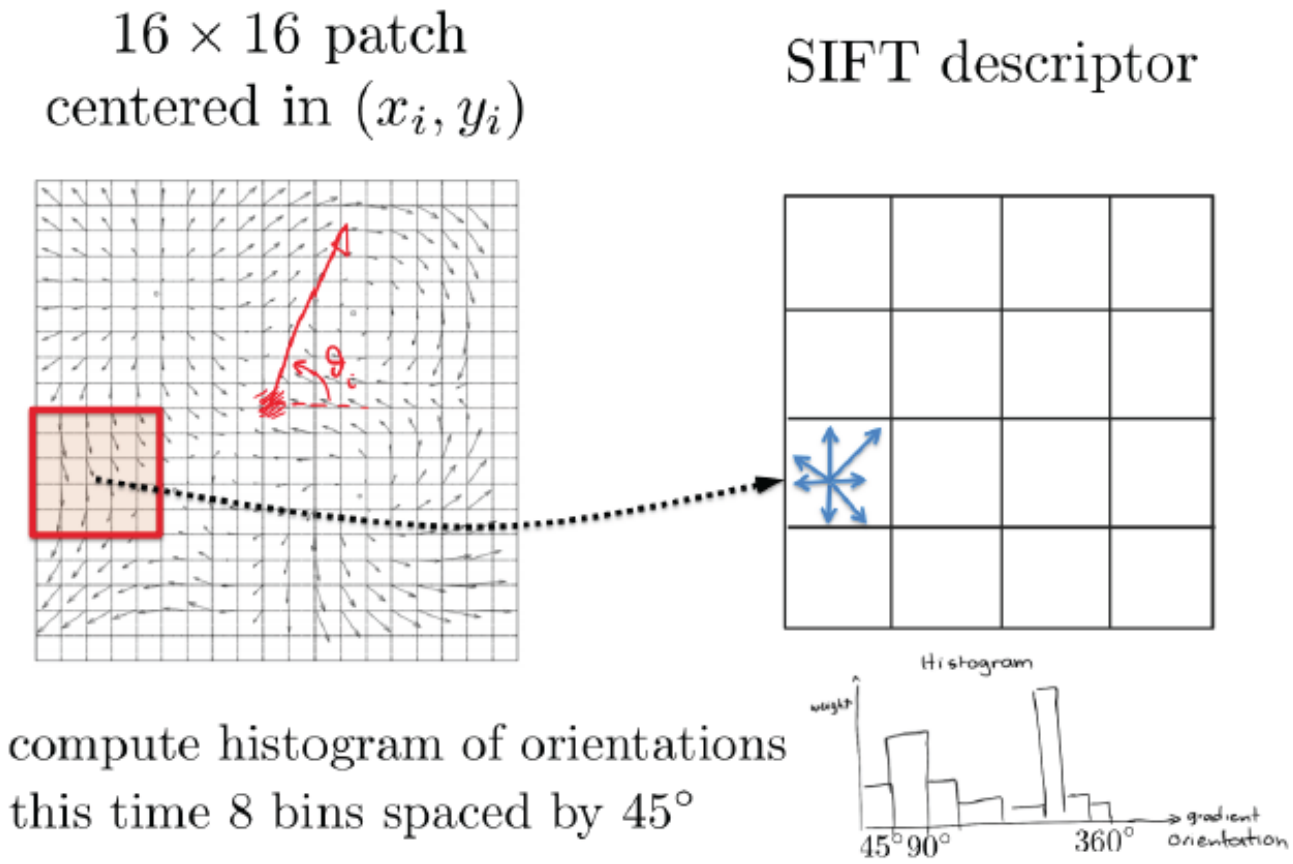




[Adopted from: F. Flores-Mangas]

SIFT Descriptor: Computing the Feature Vector

- Compute the orientations **relative** to the **dominant orientation**
- Form a 4×4 grid. For each grid cell compute a histogram of orientations for 8 orientation bins spaced apart by 45°

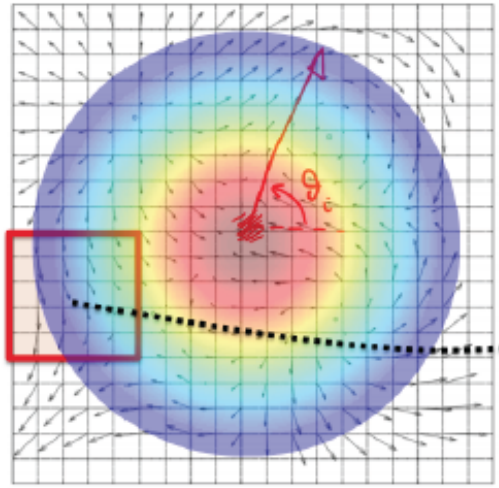


[Adopted from: F. Flores-Mangas]

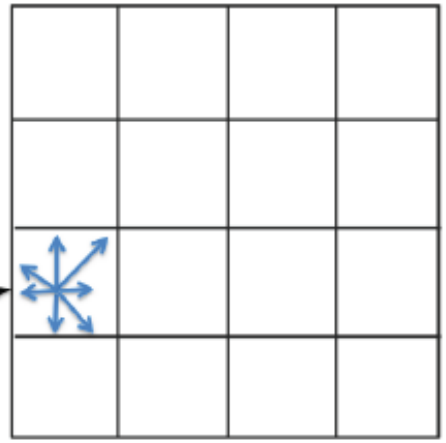
SIFT Descriptor: Computing the Feature Vector

- Compute the orientations **relative** to the **dominant orientation**
- Form a 4×4 grid. For each grid cell compute a histogram of orientations for 8 orientation bins spaced apart by 45°

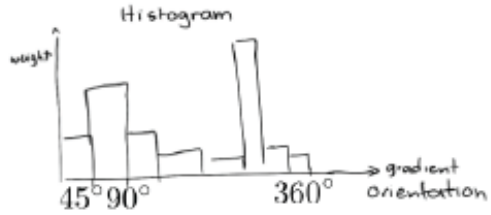
16×16 patch
centered in (x_i, y_i)



SIFT descriptor



again weigh contributions
this time: $|\nabla I(x, y)| \cdot G_{0.5\rho}$

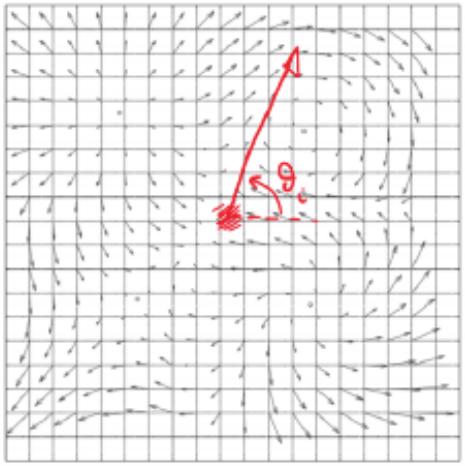


[Adopted from: F. Flores-Mangas]

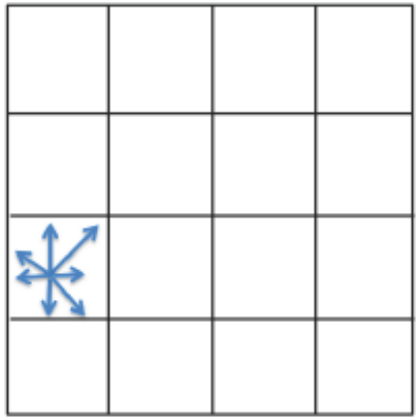
SIFT Descriptor: Computing the Feature Vector

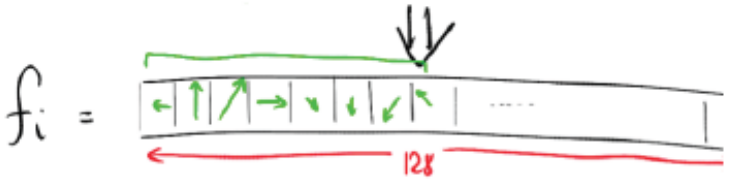
- Compute the orientations **relative** to the **dominant orientation**
- Form a 4×4 grid. For each grid cell compute a histogram of orientations for 8 orientation bins spaced apart by 45°
- Form the 128 dimensional feature vector

16×16 patch centered in (x_i, y_i)



SIFT descriptor





[Adopted from: F. Flores-Mangas]

SIFT Descriptor: Post Processing

- The resulting 128 non-negative vector is the raw version of the descriptor
- To reduce effects of contrast or gain:
  - normalize the vector to be unit norm : $f_i = f_i / \|f_i\|$
- To reduce effects of illumination variations:
  - clip values to 0.2, and renormalize to unit norm

SIFT Properties:

Invariant to:

- Scale
- Rotation

Partially invariant to:

- Illumination changes
- Camera viewpoint (up to 60 degrees out of plane rotation)
- Occlusion, clutter

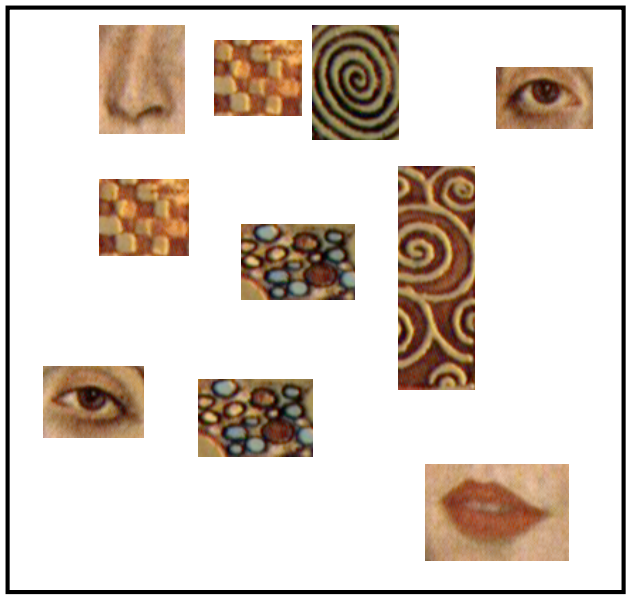
Fast and efficient

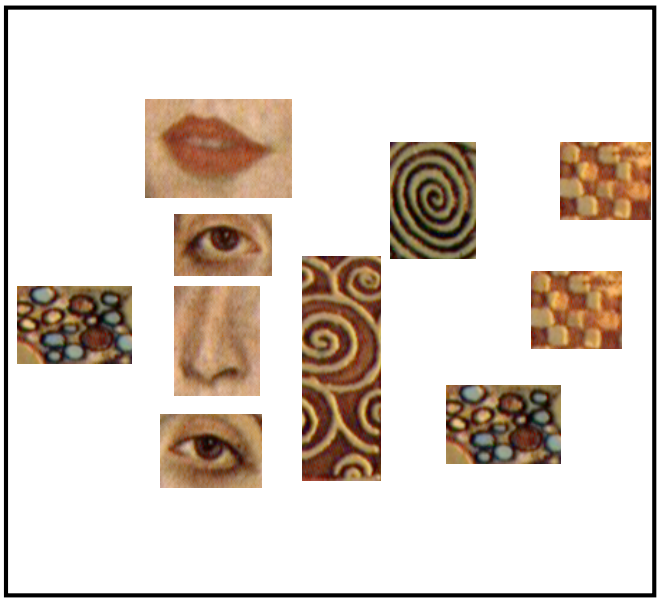
Code available

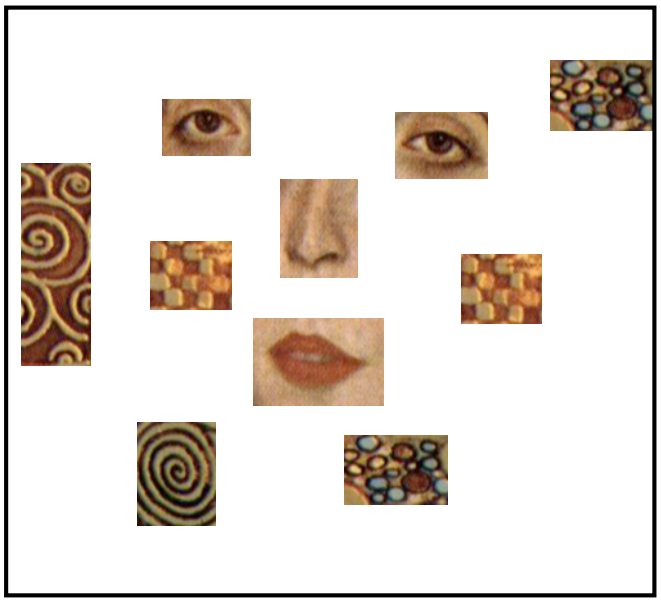
PCA-SIFT

- The dimension of the descriptor is large
- Can use PCA to reduce the dimension ~10

Problem with bag-of-words







All have equal probability for bag-of-words methods

Location information is important

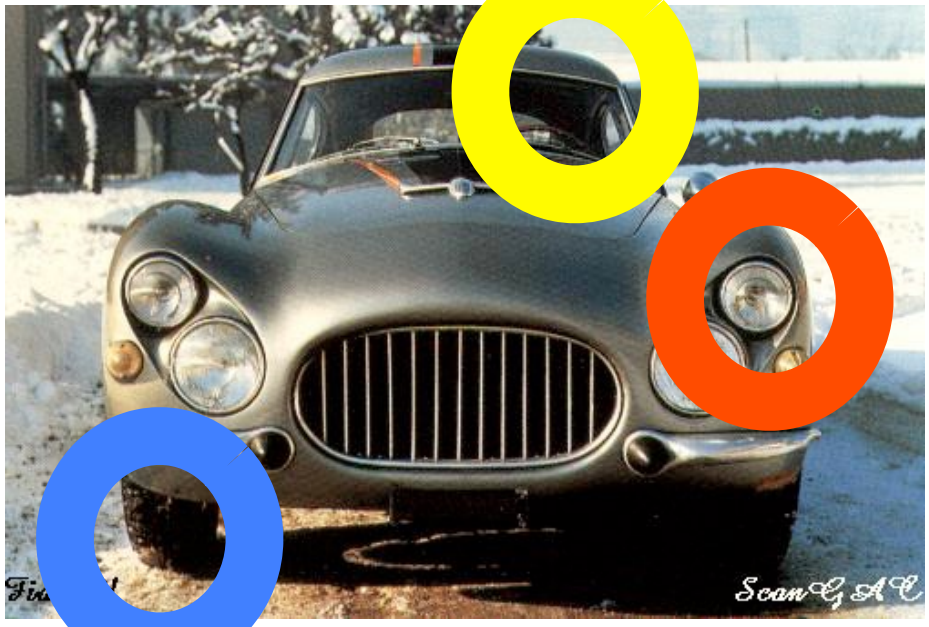
Bag of Words and Spatial Information

A bag of words throws away spatial relationships between features.

Middle ground:

- Visual “phrases”: frequently co-occurring words
- Semi-local features: describe configuration, neighborhood
- Let position be part of each feature
- Count bags of words only within sub-grids of an image
- After matching, verify spatial consistency

Parts & Structure



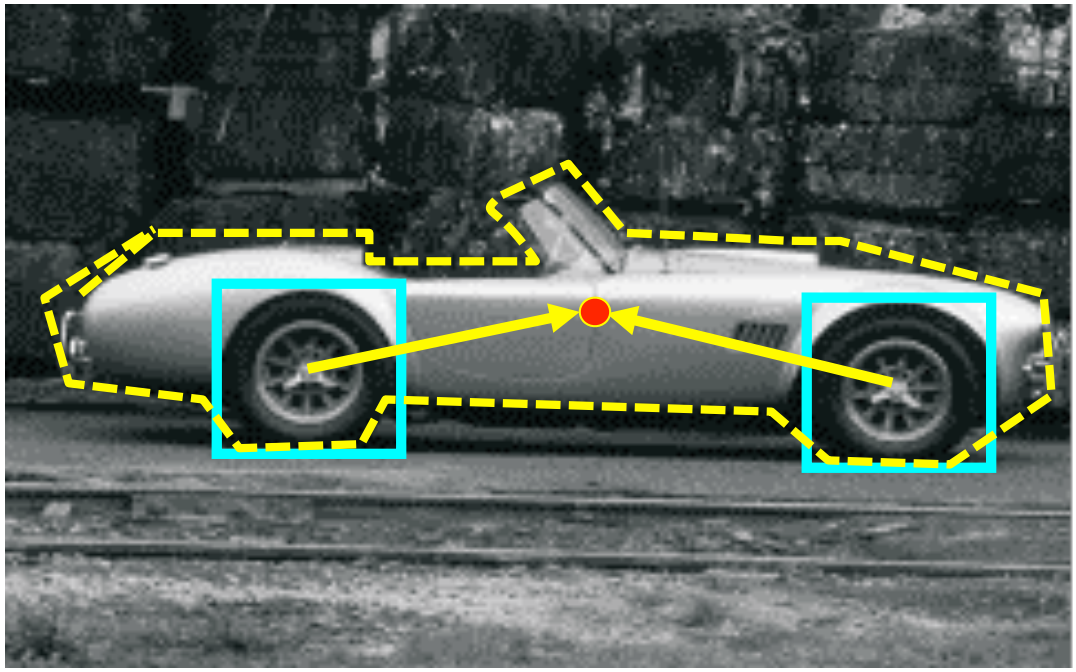
Implicit Shape Model

Use Hough Space Voting to find object:

- Learn spatial distributions of the words wrt to a “reference point”
- Use HT to vote for reference points in the test image

Implicit shape models

- Visual vocabulary is used to index votes for object position
  - [a visual word = “part”]



visual codeword with displacement vectors

training image annotated with object localization info

B. Leibe, A. Leonardis, and B. Schiele, Combined Object Categorization and Segmentation with an Implicit Shape Model, ECCV Workshop on Statistical Learning in Computer Vision 2004

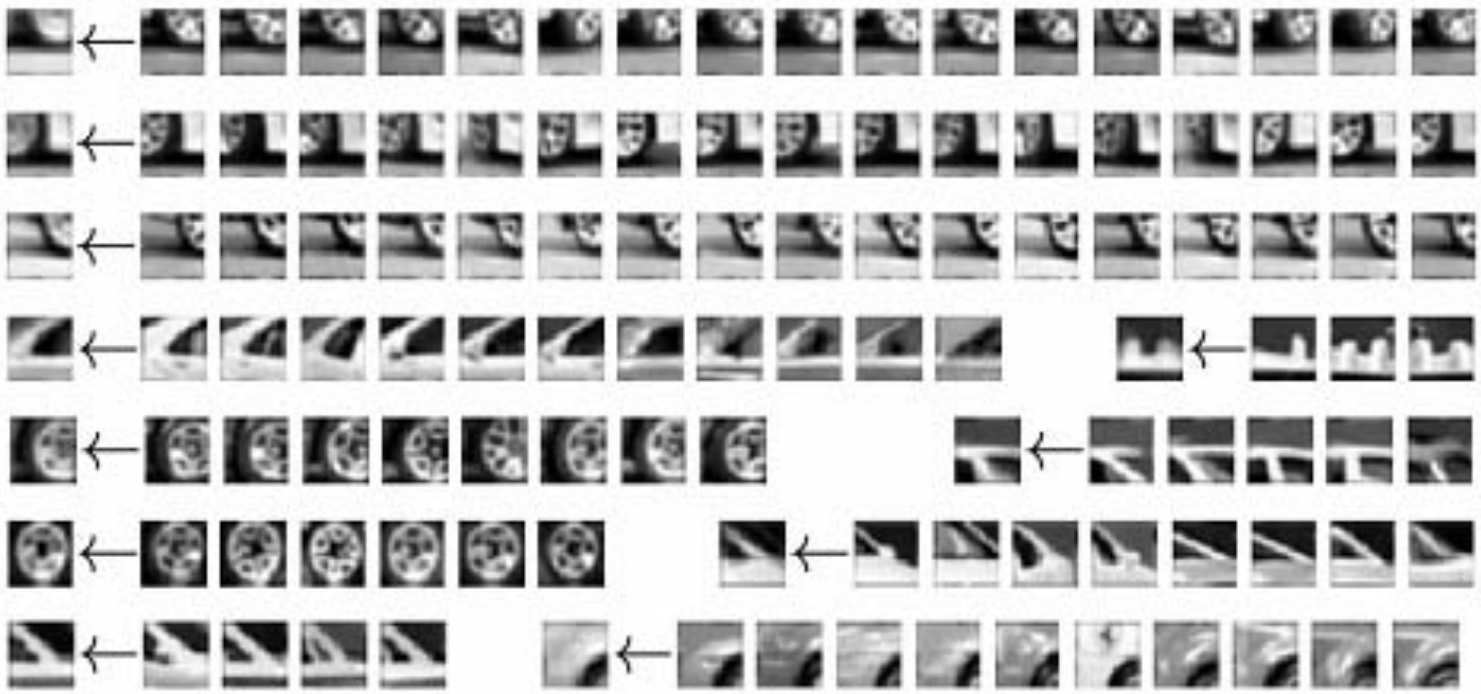
Implicit shape models

- Visual vocabulary is used to index votes for object position
  - [a visual word = “part”]

test image

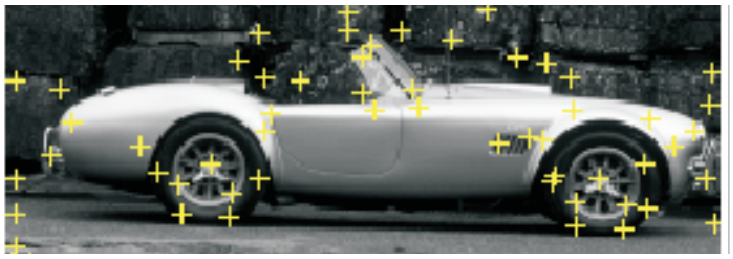
Implicit shape models: Training

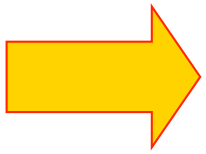
1. Build vocabulary of patches around extracted interest points using clustering

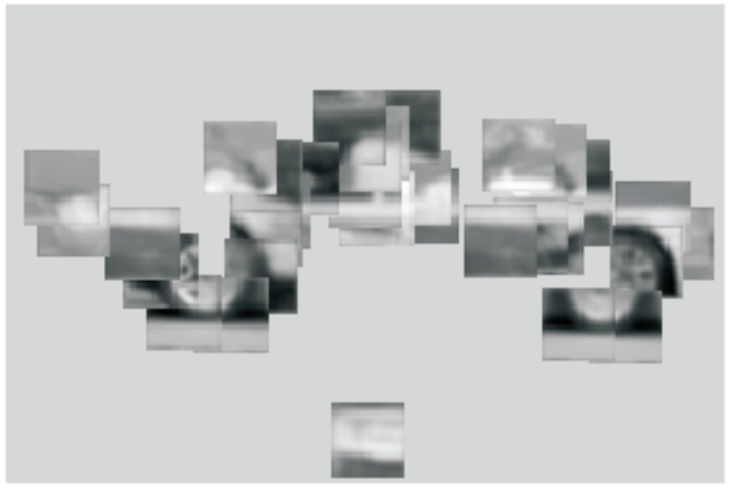


Implicit shape models: Training

1. Build vocabulary of patches around extracted interest points using clustering
2. Map the patch around each interest point to closest word

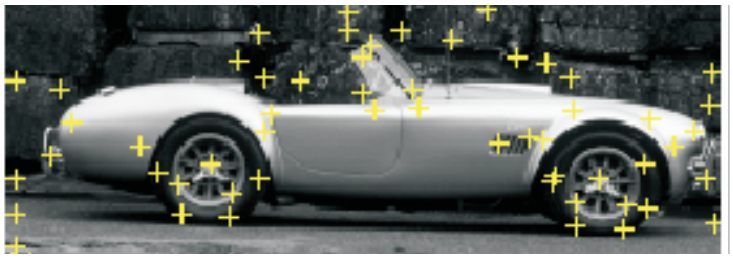


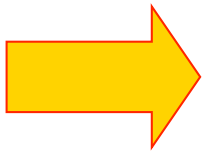


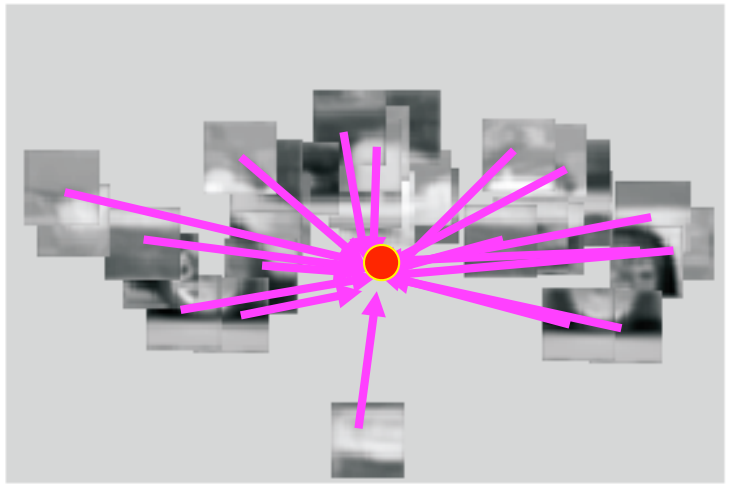


Implicit shape models: Training

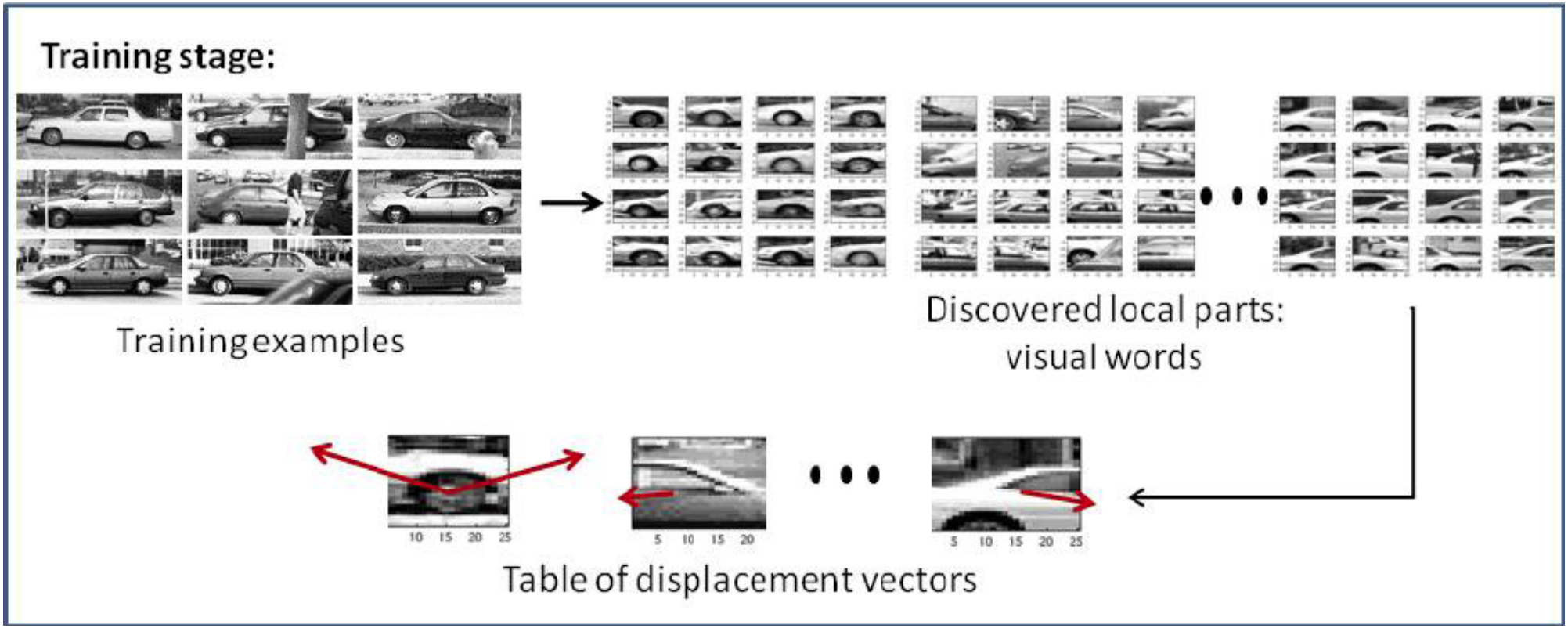
1. Build vocabulary of patches around extracted interest points using clustering
2. Map the patch around each interest point to closest word
3. For each word, store all positions it was found, relative to object center







Training


Training stage:
Training examples
Discovered local parts:
visual words
Table of displacement vectors

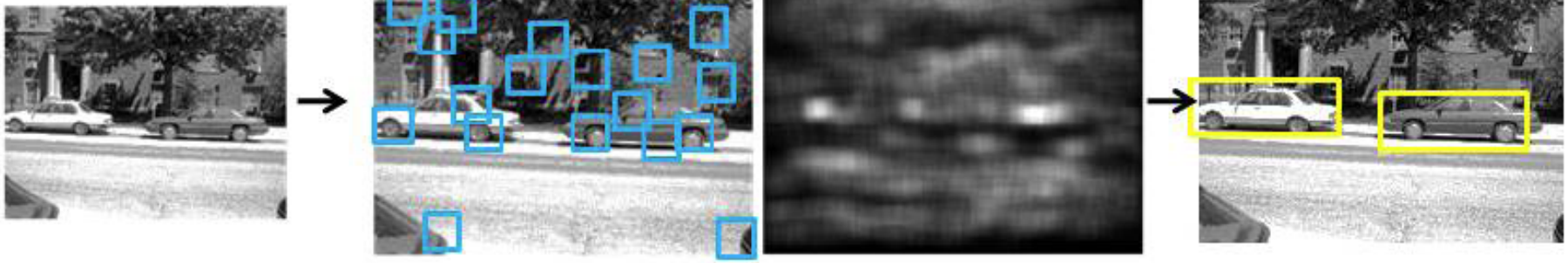
Implicit shape models: Testing

1. Given new test image, extract patches, match to vocabulary words
2. Cast votes for possible positions of object center
3. Search for maxima in voting space
4. (Extract weighted segmentation mask based on stored masks for the codebook occurrences)

What is the dimension of the Hough space?

Detecting

Detection stage:



Novel image

Voting (GHT)

Predicted object locations

Detection Results

Qualitative Performance

- Recognizes different kinds of objects
- Robust to clutter, occlusion, noise, low contrast

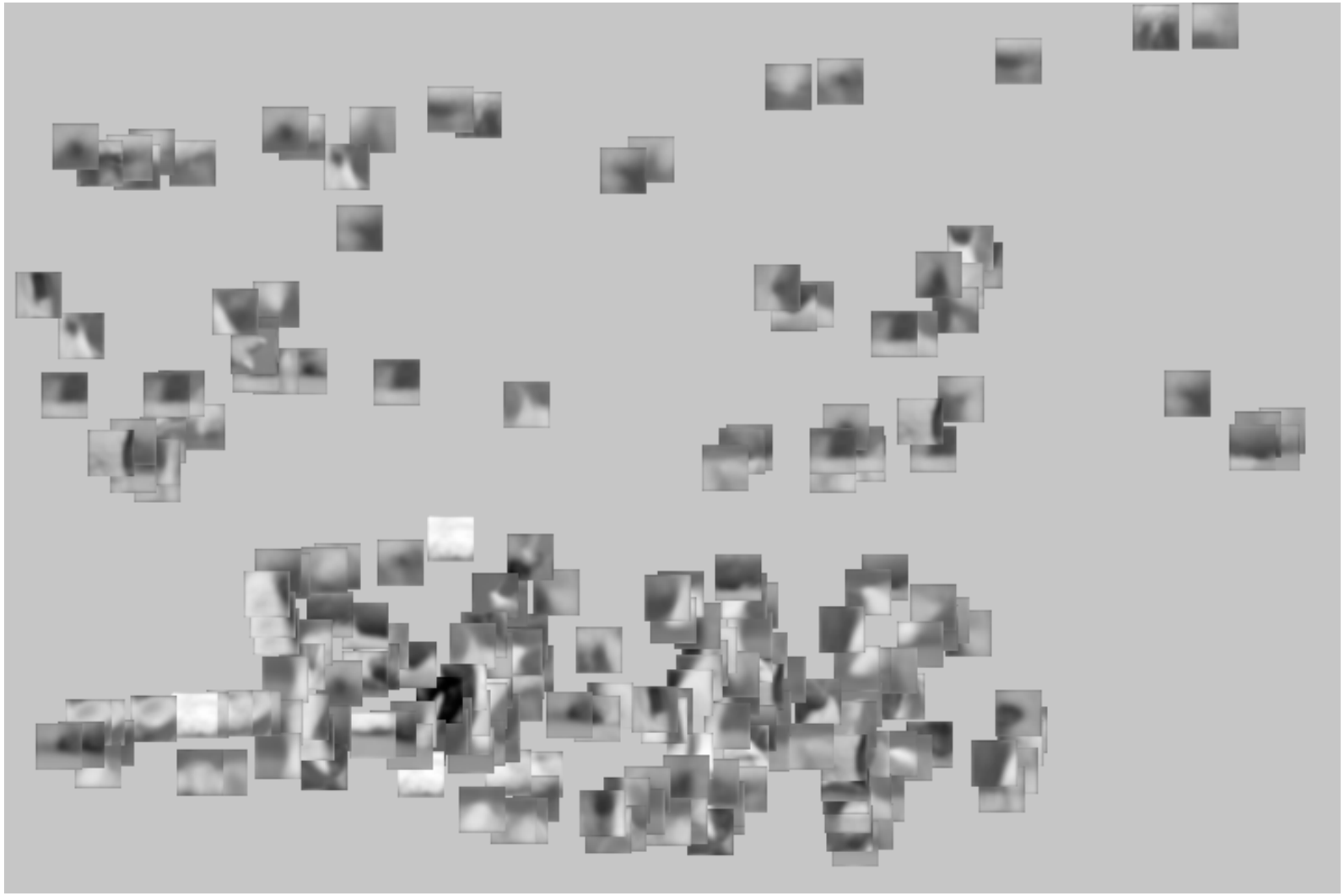
Example: Results on Cows

Original image

Example: Results on Cows

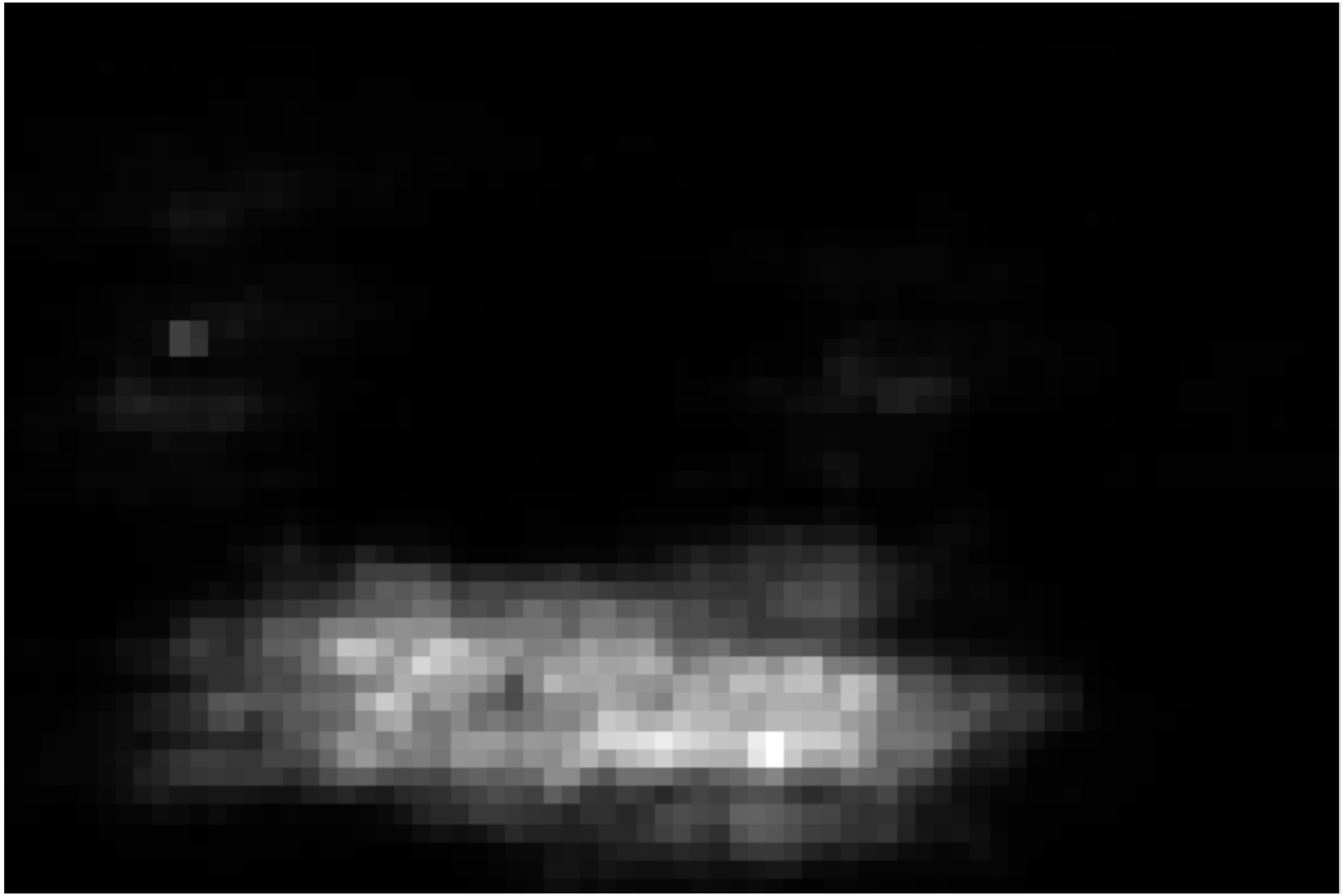
Interest points

Example: Results on Cows



Matched patches points

Example: Results on Cows



Matched **Votes**

Example: Results on Cows

1st hypothesis

Example: Results on Cows

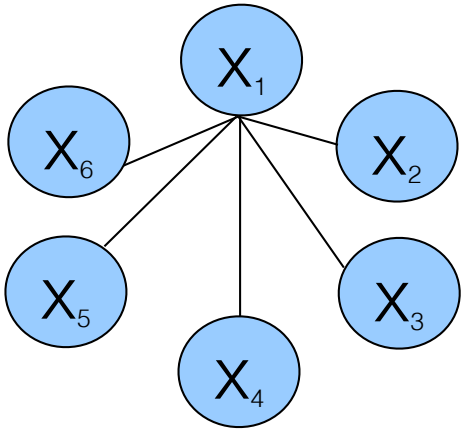
2nd hypothesis

Example: Results on Cows

3rd hypothesis

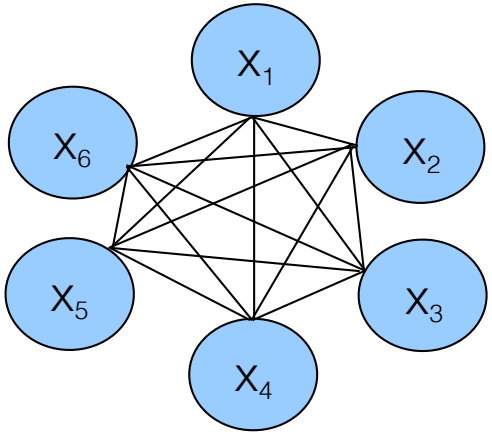
Shape representation in part-based models

"Star" shape model



- **e.g. implicit shape model**
- **Parts mutually independent**

Fully connected constellation model



- **e.g. Constellation Model**
- **Parts fully connected**

N image features, P parts in the model

Slide credit: Rob Fergus

Probabilistic constellation model

$$P(image \mid object) = P(appearance, shape \mid object)$$

Part descriptors

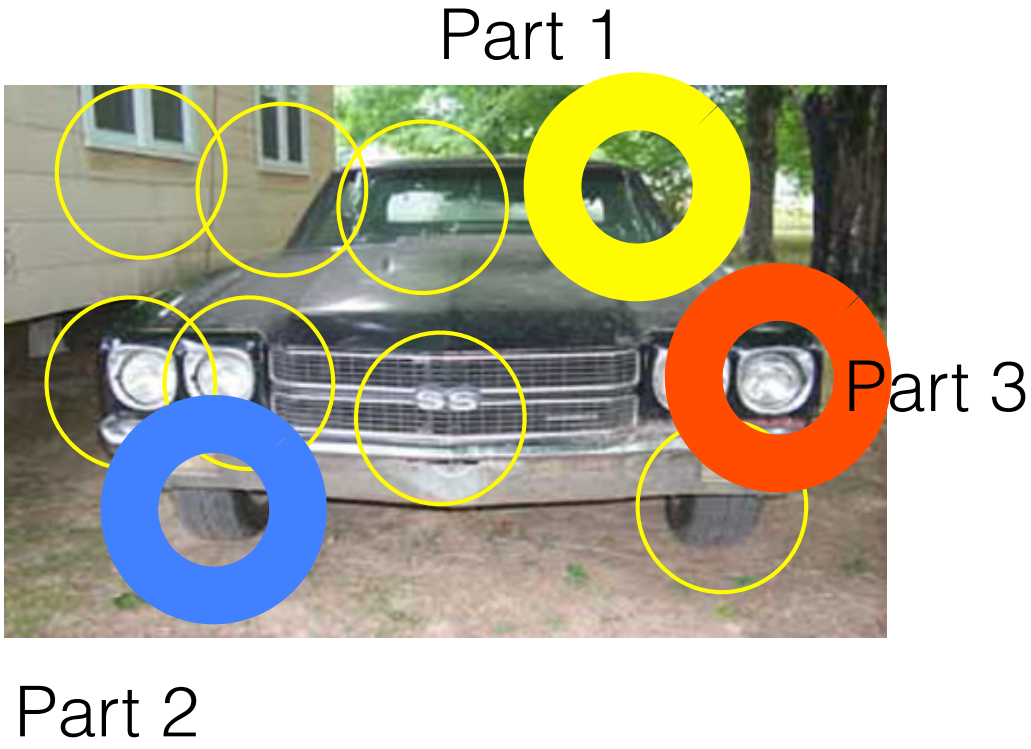
Part locations

Candidate parts

Source: Lana Lazebnik

Probabilistic constellation model

$$P(image \mid object) = P(appearance, shape \mid object)$$



Source: Lana Lazebnik

Probabilistic constellation model

$$P(image \mid object) = P(appearance, shape \mid object)$$
$$= \max_h P(appearance \mid h, object)\, p(shape \mid h, object)\, p(h \mid object)$$

h: assignment of features to parts

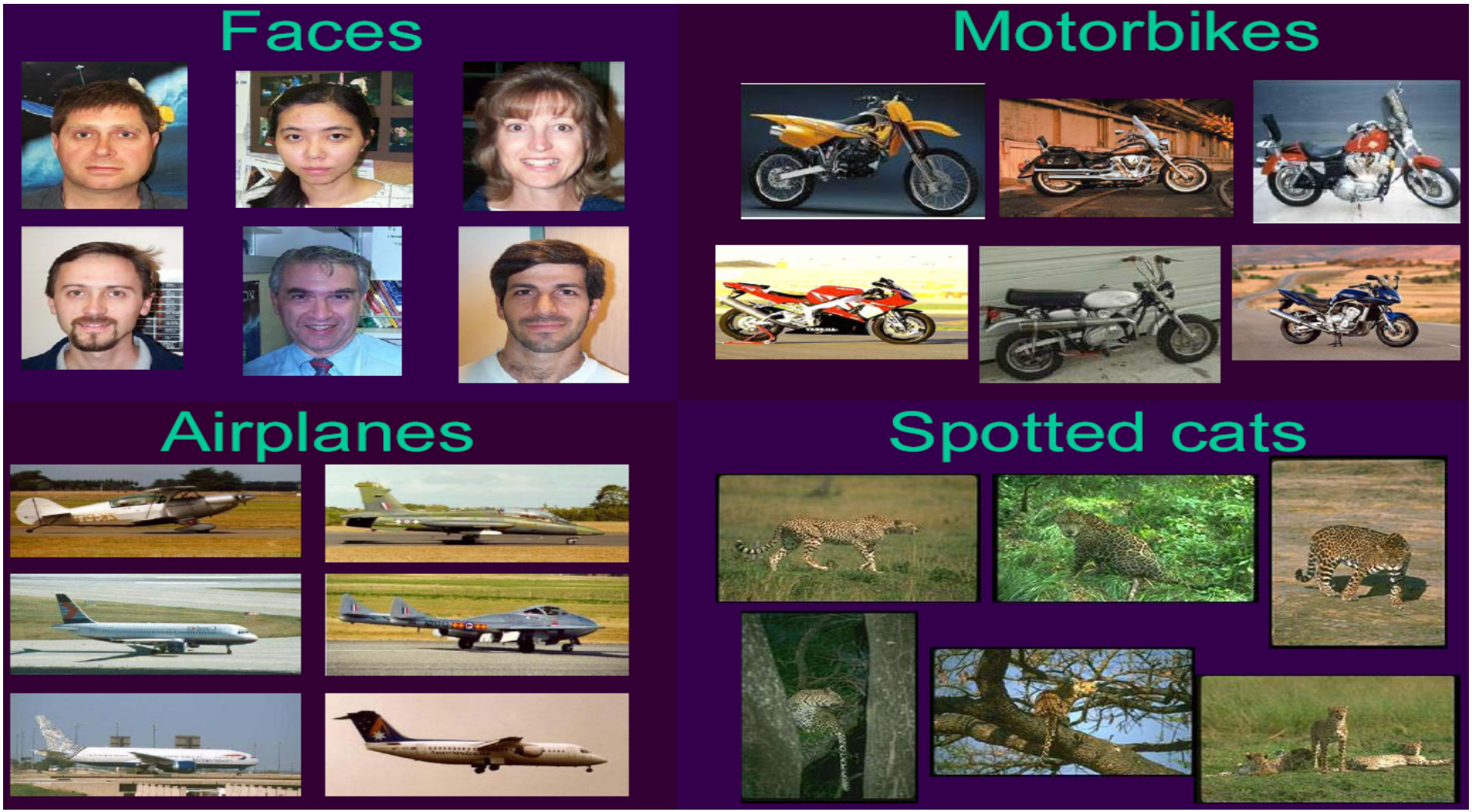
Part 1

Part 3

Part 2

Source: Lana Lazebnik

Example results from constellation model: data from four categories



Slide from Li Fei-Fei http://www.vision.caltech.edu/feifeili/Resume.htm

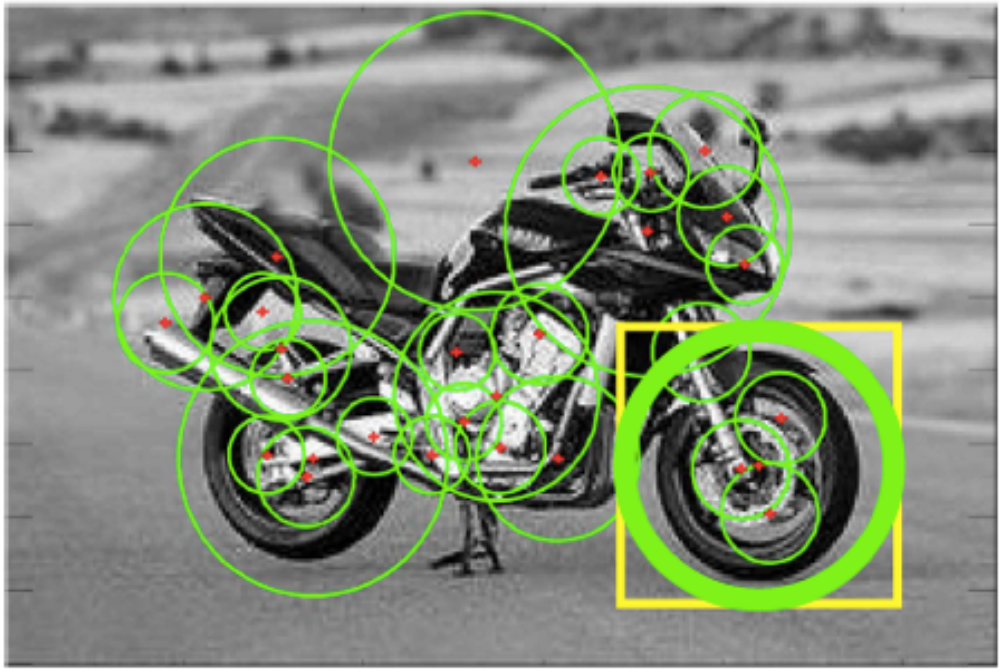
Weakly Supervised Class Learning

Train class without segmentation

Training images have the object of interest (once)

but do not know where or the scale

Semi-supervised Training (Fergus et al '03)



- Find regions within image
- Use salient region operator (Kadir & Brady 01)

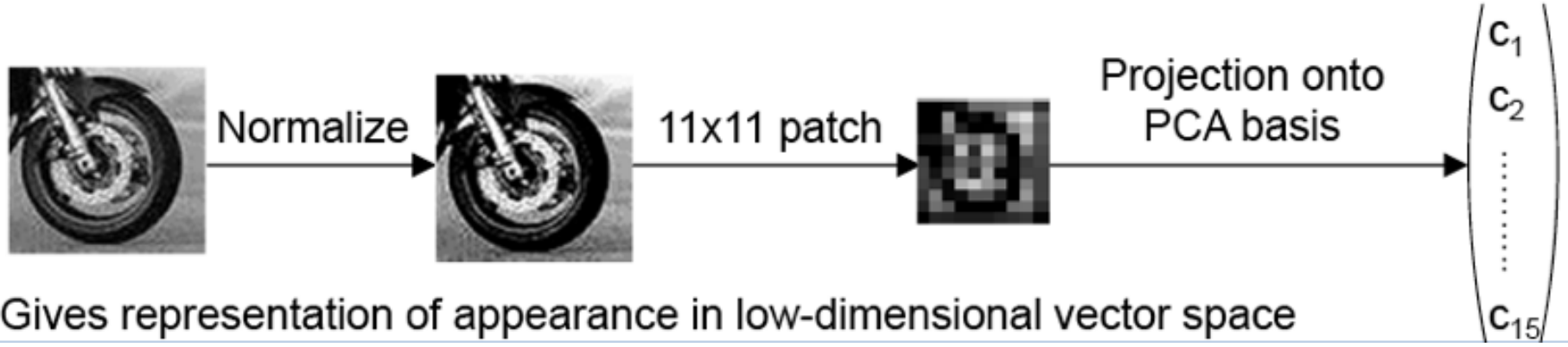
Location

(x,y) coords. of region centre

Scale

Radius of region (pixels)

Appearance



Gives representation of appearance in low-dimensional vector space

Motorcycle Example

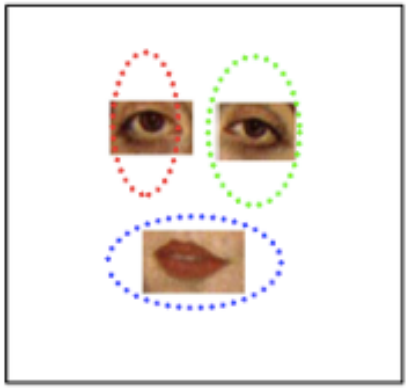
•Kadir & Brady saliency region detector

Generative Model

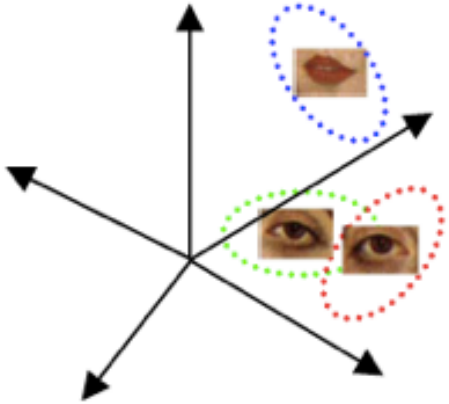
Foreground model

based on Burl, Weber et al. [ECCV '98, '00]

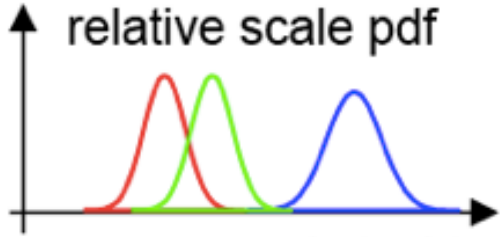
Gaussian shape pdf



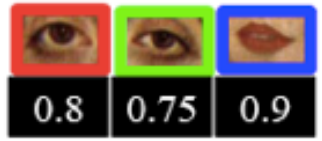
Gaussian part appearance pdf



Gaussian relative scale pdf



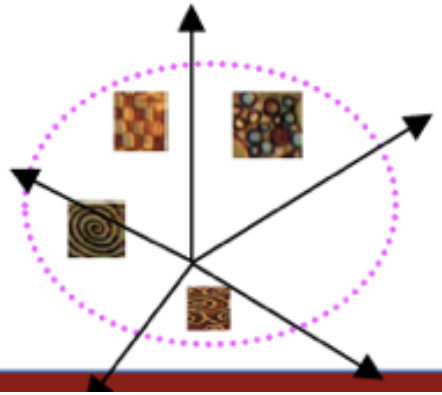
Prob. of detection



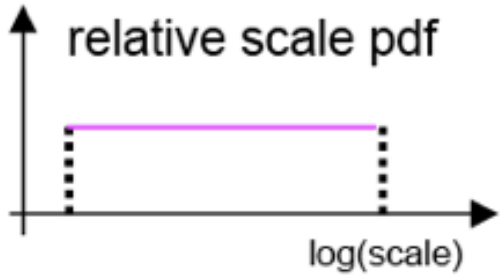
Clutter model

Uniform shape pdf

Gaussian background appearance pdf



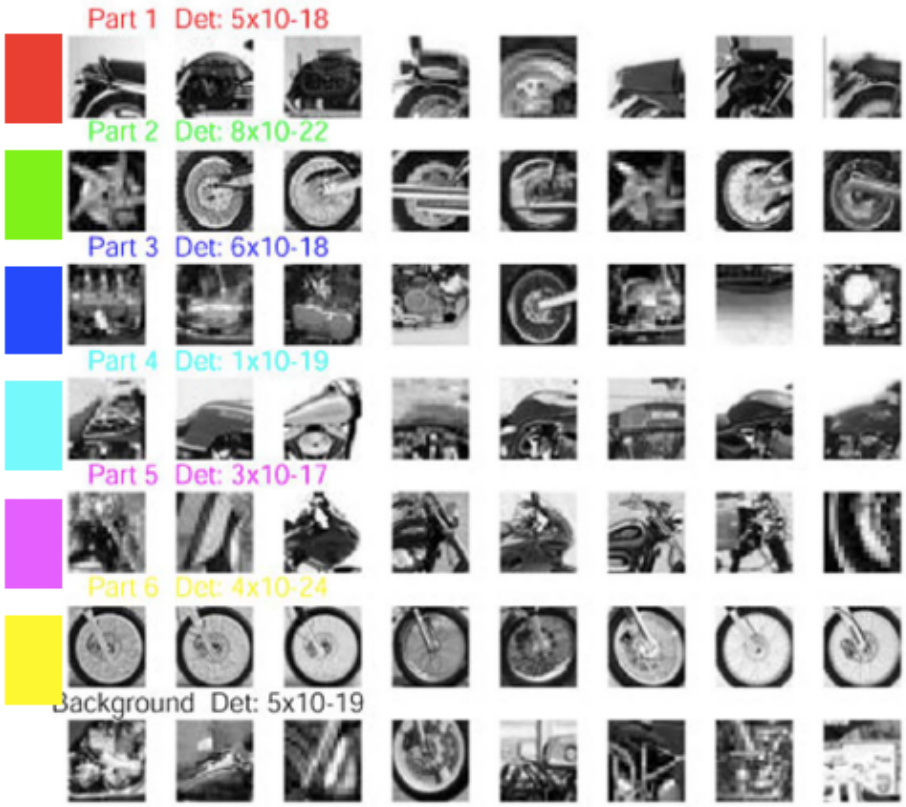
Uniform relative scale pdf



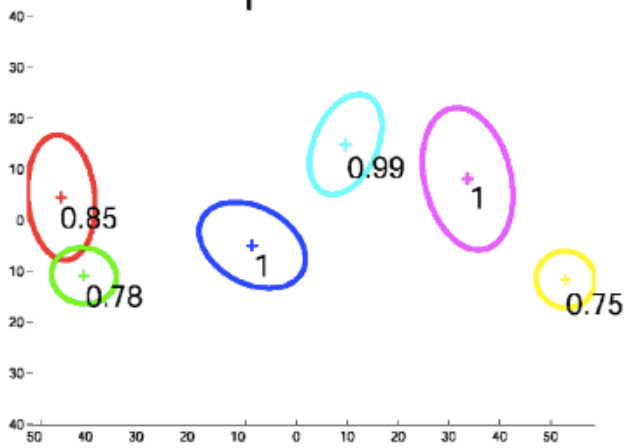
Poission pdf on # detections

Motorbikes

Samples from appearance model

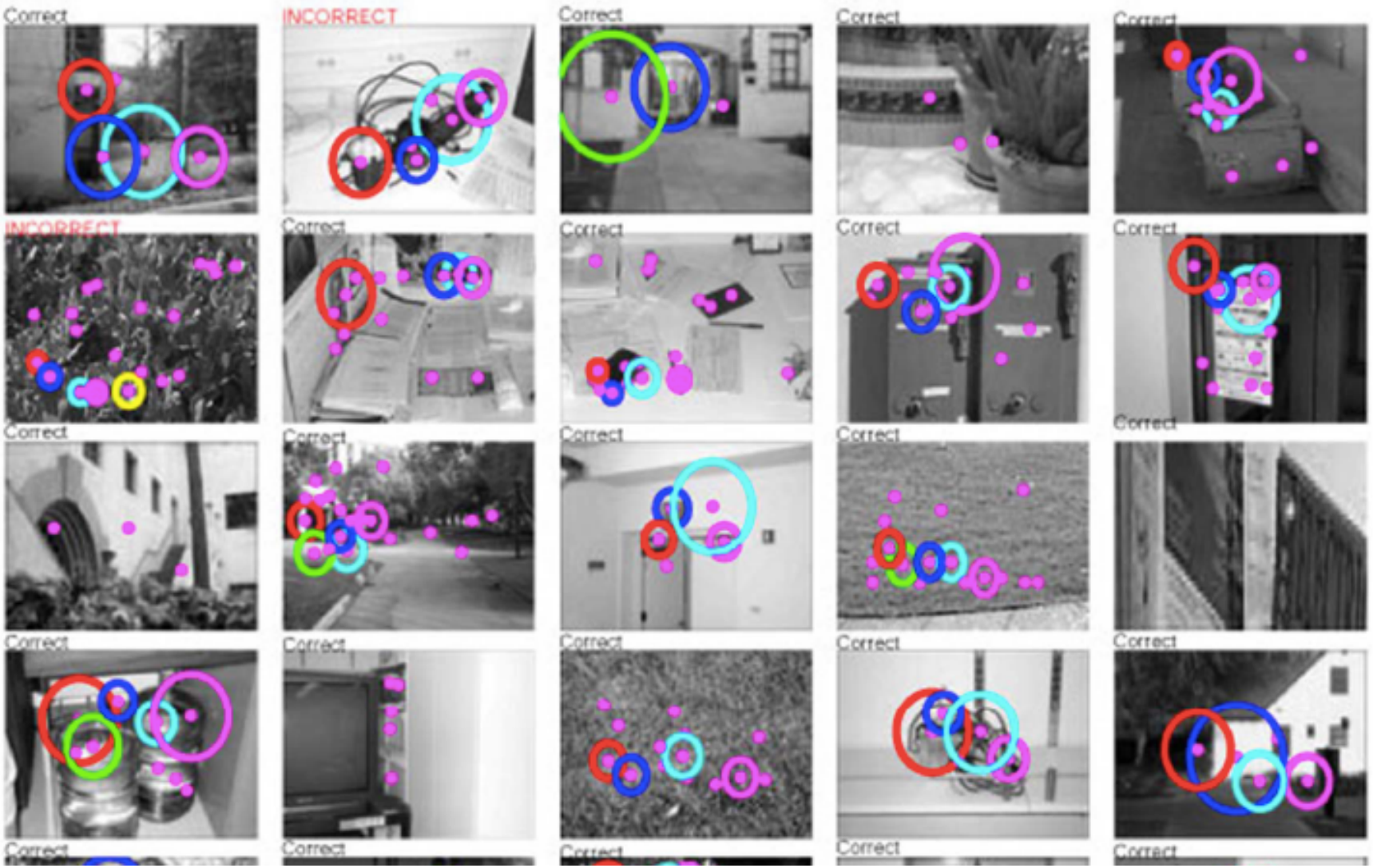


Shape model

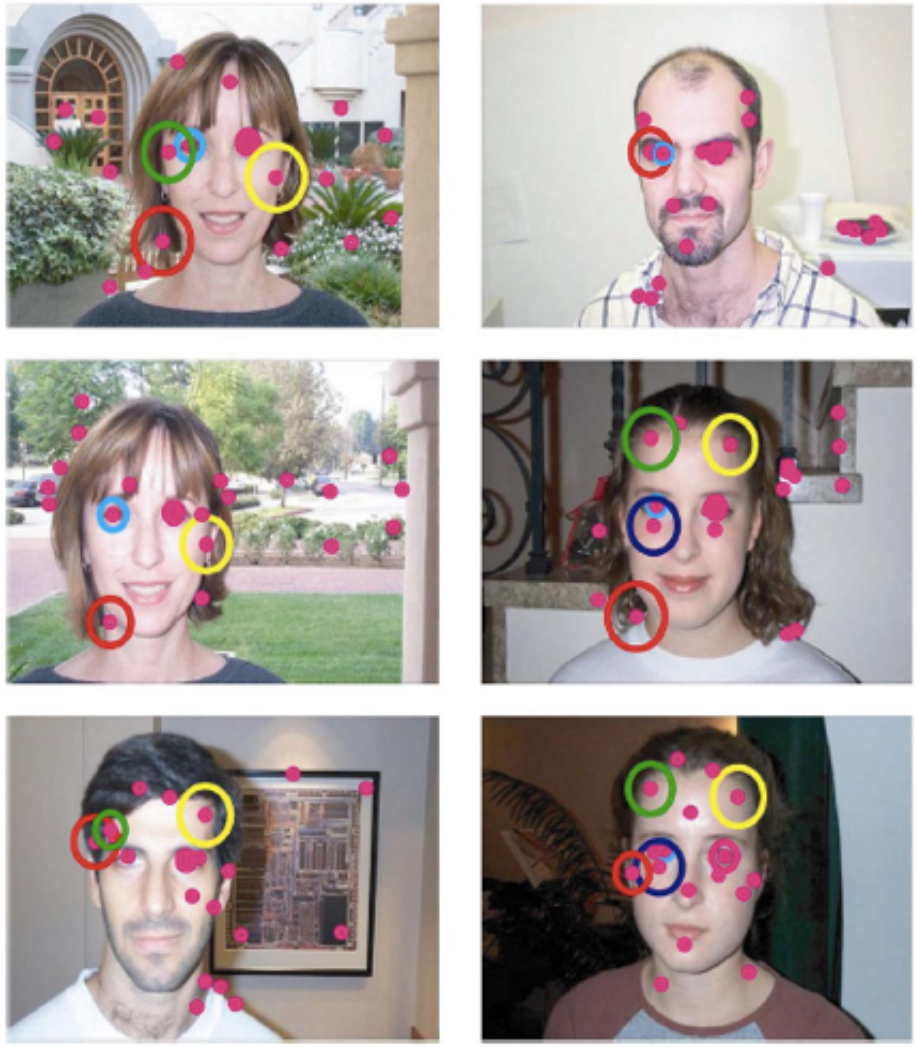


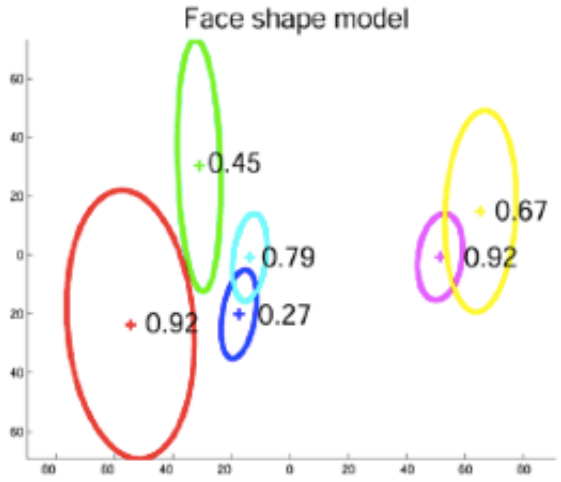
Test images: size of circles indicates score of hypothesis

Motorbike Model on Background Images

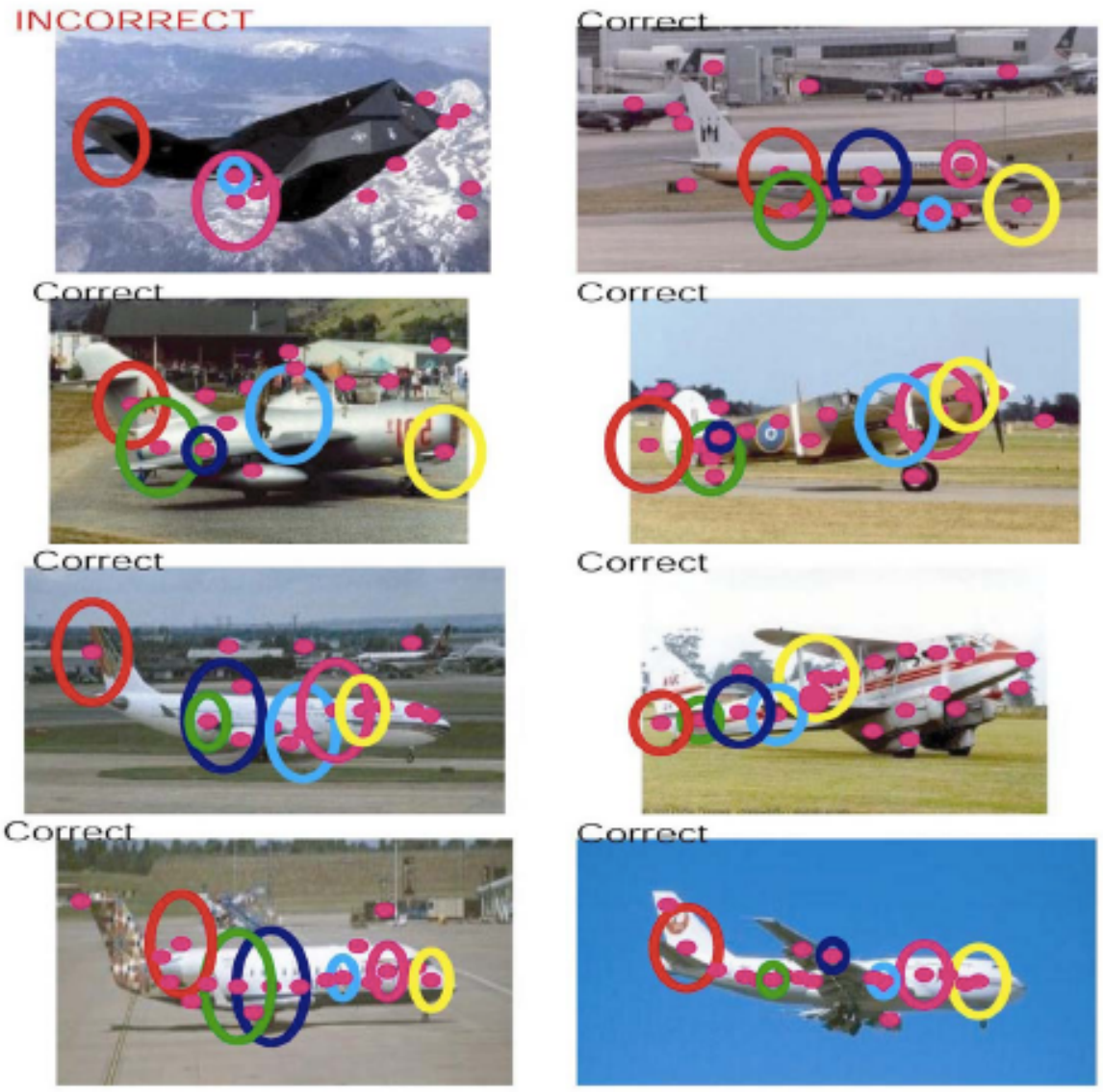


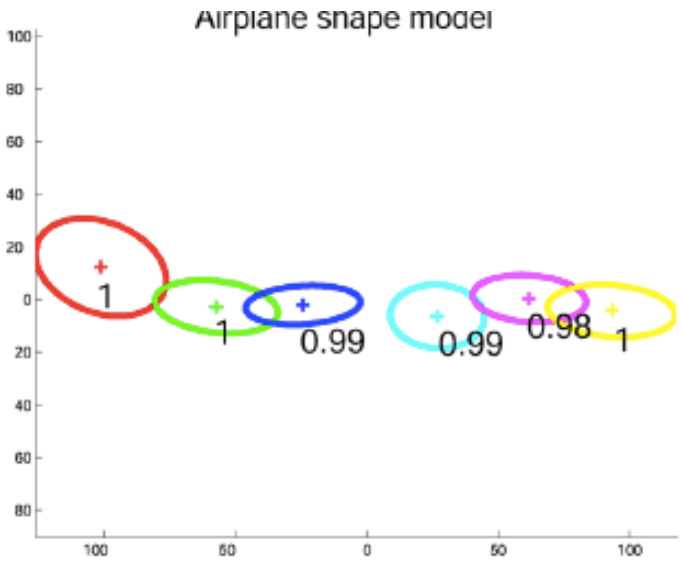
Frontal Faces

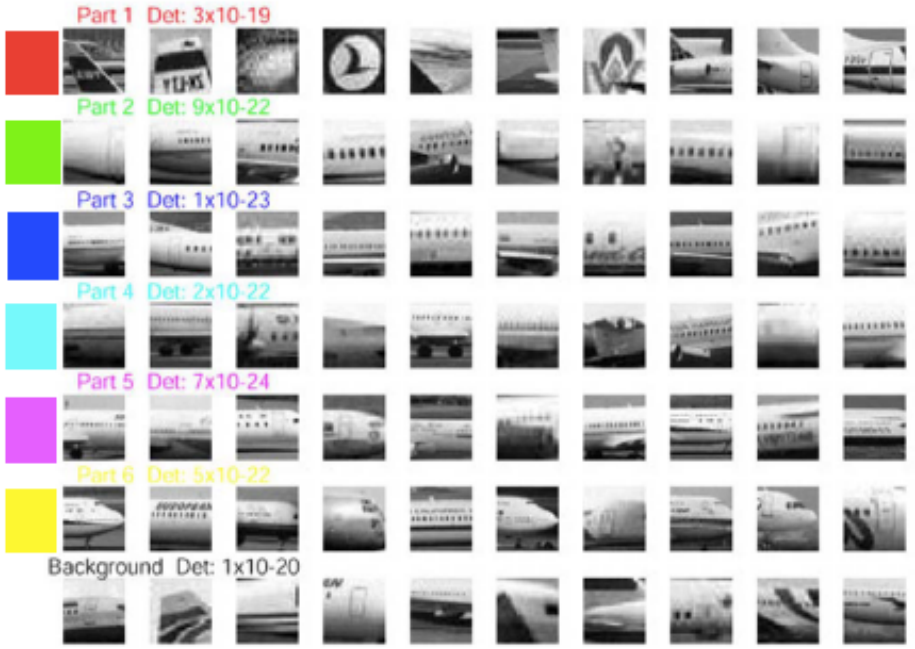




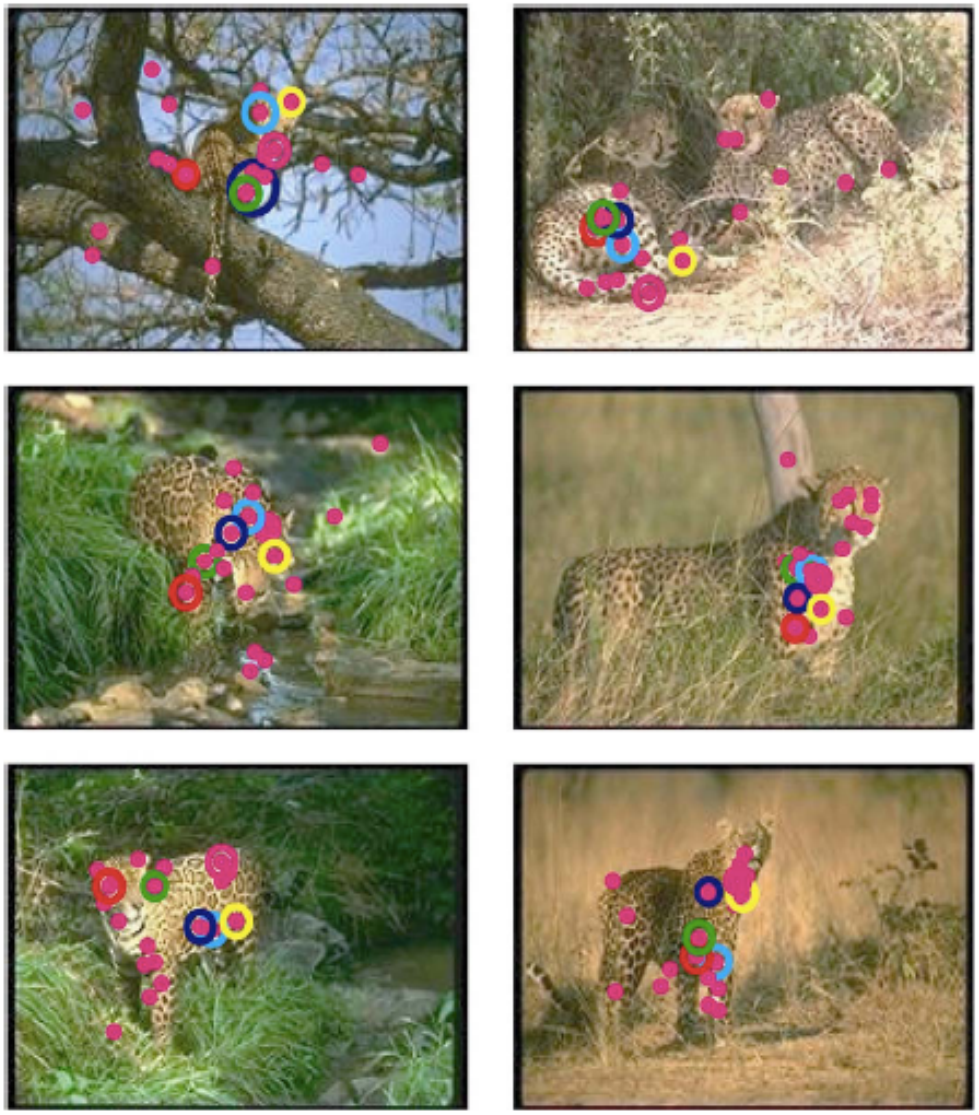
Airplanes

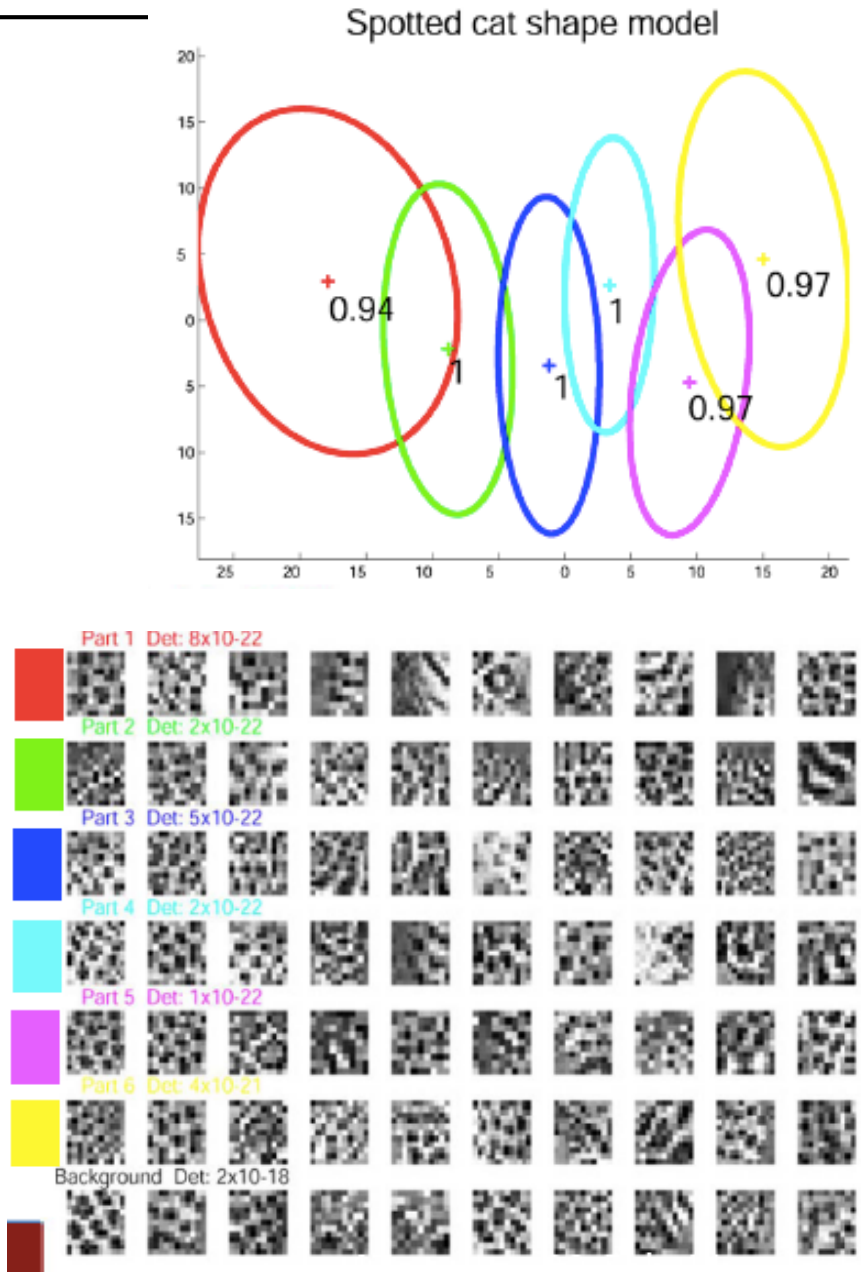

INCORRECT
Correct
Correct
Correct
Correct
Correct
Correct
Correct


Airplane shape model
100
80
60
40
20
0
20
40
60
80
100
50
0
50
100
1
1
0.99
0.99
0.98
1


Part 1 Det: 3x10-19
Part 2 Det: 9x10-22
Part 3 Det: 1x10-23
Part 4 Det: 2x10-22
Part 5 Det: 7x10-24
Part 6 Det: 5x10-22
Background Det: 1x10-20

Spotted Cats





Comparison



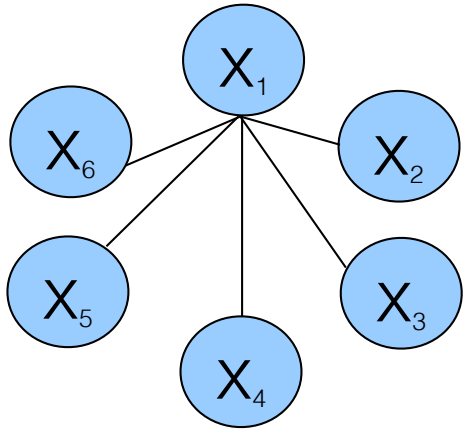



class	bag of features	bag of features	Part-based model
	Zhang et al. (2005)	Willamowski et al. (2004)	Fergus et al. (2003)
airplanes	**98.8**	97.1	90.2
cars (rear)	98.3	**98.6**	90.3
cars (side)	**95.0**	87.3	88.5
faces	**100**	99.3	96.4
motorbikes	**98.5**	98.0	92.5
spotted cats	**97.0**	—	90.0

Source: Lana La

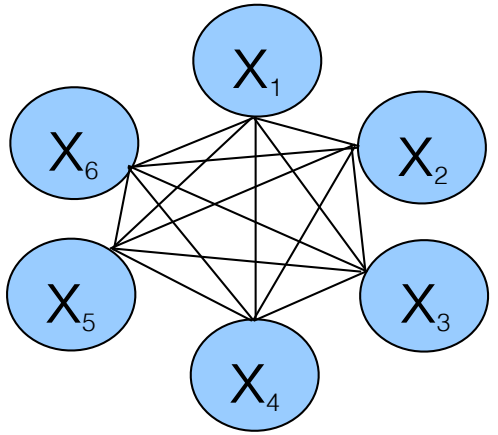
Shape representation in part-based models

"Star" shape model



- **e.g. implicit shape model**
- **Parts mutually independent**
- **Recognition complexity: O(NP)**
- **Method: Gen. Hough Transform**

Fully connected constellation model



- **e.g. Constellation Model**
- **Parts fully connected**
- **Recognition complexity: O(N^P)**
- **Method: Exhaustive search**

N image features, P parts in the model

Slide credit: Rob Fergus

Summary:
part-based and local feature models for generic object recognition

Histograms of visual words to capture global or local layout in the bag-of-words framework

- Powerful in practice for image recognition

Part-based models encode category's part appearance together with 2d layout and allow detection within cluttered image

- **"implicit shape model"**: shape based on layout of all parts relative to a reference part; Generalized Hough for detection
- **"constellation model"**: explicitly model mutual spatial layout between all pairs of parts; exhaustive search for best fit of features to parts